USING MULTIPLE AGENTS IN UNCERTAINTY MINIMIZATION OF ABLATING TARGET SOURCES

A Dissertation
Presented to
The Academic Faculty

By

Richard A. Coogle

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2014

USING MULTIPLE AGENTS IN UNCERTAINTY MINIMIZATION OF ABLATING TARGET SOURCES

Approved by:

Dr. Ayanna M. Howard, Committee Chair
Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Mary Ann Weitnauer
Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ayanna M. Howard, Advisor
Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Thomas R. Collins
Principal Research Engineer
Georgia Tech Research Institute

Dr. Magnus Egerstedt
Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ronald Arkin
Professor, College of Computing
Georgia Institute of Technology

Date Approved: August 2014

For my family.

ACKNOWLEDGMENTS

While one person must eventually write the text of a doctoral thesis, what brings them to the point of completing such a task is generally a large and often colorful cast of characters.

While I would love to write an individual note to all of you, that would likely make this acknowledgements section longer than the actual thesis. So consider this my apology, while I hit the highlights.

I would like to thank first and foremost my family for their support when I decided to embark on this "final" adventure at Georgia Tech. Without your love and encouragement, I am not sure I would have made it this far. Nor would I have even begun it if you had not instilled in me from a very early age a boundless need for knowledge and a boundless want for understanding the world around us.

Of course, I would also like to thank Dr. Ayanna Howard, my advisor, for taking a chance on me, for guiding me through my research, for being extremely patient with me (especially when it came to my absent-minded tendencies), and for helping to untangle the issues that came up here and there. I used to be the so-called "model student" once, I promise.

I would also like to thank my dissertation committee for contributing their time to evaluating my thesis and for their valuable input that strengthened it.

In addition, the support and camaraderie of my colleagues in Dr. Howard's Human-Automation Systems (HumAnS) Lab and my colleagues at the Georgia Tech Research Institute (GTRI) has been invaluable. Thanks for simply being there and being (and continue being) awesome.

Finally, and hardily the least, thank you to all my friends who have had to put up with me during this entire adventure. Your love and support has been fantastic.

Undertaking this endeavor can be easily compared to the classic story archetype of the Hero's Journey. Consider this a slain dragon.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The objective of this research effort is to provide an efficient methodology for a multi-agent robotic system to observe moving targets that are generated from an ablation process. An ablation process is a process where a larger mass is reduced in volume as a result of erosion; this erosion results in smaller, independent masses. An example of such a process is the natural process that gives rise to icebergs, which are generated through an ablation process referred to as ice calving.

Ships that operate in polar regions continue to face the threat of floating ice sheets and icebergs generated from the ice-ablation process. Although systems have been implemented to track these threats with varying degrees of success, many of these techniques require that the operations are conducted outside of some boundary where the icebergs are known not to drift. Since instances where polar operations must be conducted within such a boundary line do exist (*e.g.*, resource exploration), methods for situational awareness of icebergs for these operations are necessary. In this research, efficacy of these methods is correlated to the initial acquisition time of observing newly ablated targets, as it provides for the ability to enact early countermeasures.

To address the research objective, the iceberg tracking problem is defined such that it is re-cast within a class of robotic, multi-agent, target-observation problems. From this new definition, the primary contributions of this research are obtained: 1) A definition of the iceberg observation problem that extends an existing robotic observation problem to the unique requirements for the observation of floating ice masses; 2) A method for modeling the activity regions on an ablating source to extract ideal search regions to quickly acquire newly ablated targets; and 3) A method for extracting metrics for this model that can be used to assess performance of observation algorithms and perform resource allocation. A robot controller is developed that implements the algorithms that result from these contributions, uses methods for using multiple robotic agents to observe a region according to

the modeling methodogy, and performs reallocation of observation resources as the model changes. Comparisons are made to existing target acquisition techniques.

CHAPTER 1

INTRODUCTION

The danger of collision with floating ice is a threat that continues to exist for polar, ship-based operations [1]. Organizations and systems have been implemented to track and provide information about these threats. The most well-known and perhaps important of these organizations is the International Ice Patrol (IIP) [2], a multinational coalition dedicated to tracking iceberg threats in the northern Atlantic Ocean. They provide data products that establish an iceberg boundary or "iceberg limit," outside of which a ship is safe from collision with icebergs. However, for polar operations such as resource exploration and mining, operations within the iceberg limit may be necessary.

Collision avoidance takes an amount of advance planning by the ship's crew, as a result of the slow speeds and lack of mobility of the craft involved. Oil platforms have no mobility to speak of. Therefore, a method of sensing and tracking floating ice is necessary for situational awareness. Usually, the primary means available to build such avoidance plans is derived from radar and visual observations. Unfortunately, disadvantages exist for both radar and visual observations [3]. Smaller icebergs, for example, can be difficult for a radar to track because of their low-magnitude radar cross sections (RCS), confusion with sea-clutter in stormy situations, and the fact that they typically are not detected in satellite imagery as a result of their size [4]. These smaller icebergs, which range from the size of a car to the interior of a small house, are generally a greater threat than the larger ones, as a result of the fact that they present only a small portion of their full mass above water, and may not be detected until it is too late. In addition to the limitations on detecting smaller icebergs, satellite-based synthetic aperture radar (SAR) imaging is generally not available in real-time. Aircraft radar sweeps can be expensive and cannot be continously conducted. Visual observation requires that a dedicated crew continuously monitors for new threats. Furthermore, ice threats are not completely observable in this manner, and

once a threat is observed, depending on the type of ship, not enough time will exist for the crew of the ship to begin evasive maneuvers. Additionally, for the case of an oil platform, other countermeasures would have to be used. Therefore, a method of remotely sensing these threats with a minimized emphasis on radar sensing may be the best approach to the problem, with a focus on understanding the sources of these threats.

Floating ice masses are generated by an *ice-ablation* process called *ice calving* [5]. Depending on temperature, glacial-flow rate, and the base strength of their source glaciers, these ice masses have varying rates of iceberg and floating-ice generation. For example, Figure 1 is a diagram of the flow rates of Antarctic glaciers, which vary depending on the proximity of the glacier to the coast and other factors. The diagram shows that the flow rate is not the same at different interface regions of the ice with the ocean. Indeed, in some cases, the flow rate is much higher at some regions than at others. These high-activity regions in the figure are the regions that fade in to a faint green color from the yellow. The iceberg calving rate of these ice masses would be higher at the regions with a greater flow rate at the glacier-sea interface. However, this information might either not be immediately available to an observer or difficult to accurately interpret. The flow rate might be higher at these regions, but that does not guarantee a higher calving rate. In such cases, observations of the calved ice masses would be a more appropriate means of determining which ablating sources should have a greater monitoring emphasis.

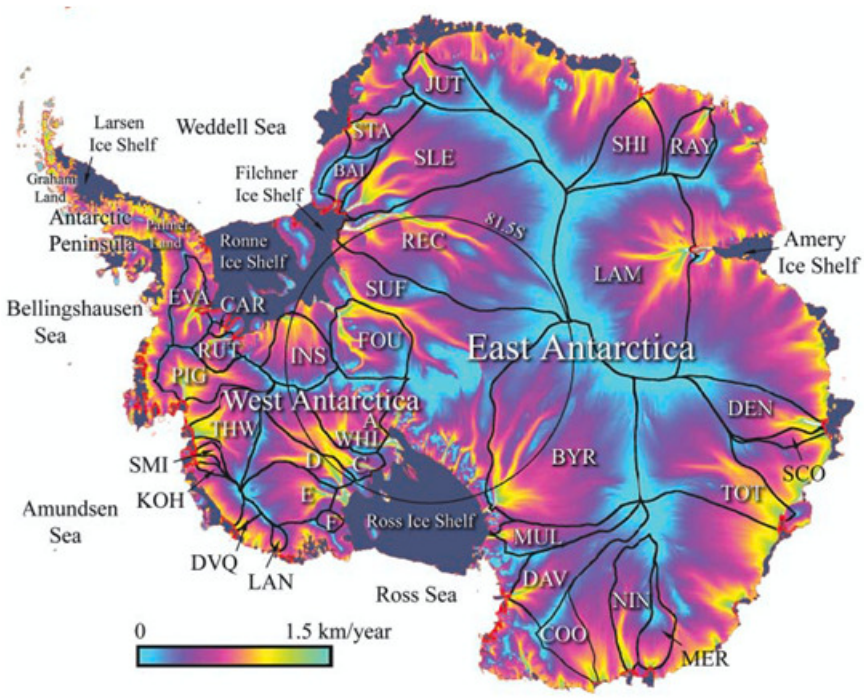


Figure 1. Flow rates of Antarctic glaciers. Image courtesy of NASA.

Based on observations of a floating ice source and the masses calved from it, a model can be developed for the locations on the source where new ice masses are generated with highest probability. Using this information, sensor resources can be reallocated to place a greater focus on the sources that generate more targets. In this case, a multiagent, robotic observation system is proposed to act as the sensor system, based on an existing, robotic, sensing framework problem.

To enable an efficient solution to this observation problem, the specific contributions of this research are as follows:

- Contribution 1: A definition of the iceberg observation problem that extends an existing robotic observation problem to the unique requirements for the observation of floating ice masses.

- Contribution 2: A probabilistic method for *in-situ* modeling of the expected locations of regions of activity on an ablating target source, so that observation resources can

be retasked to these regions.

- Contribution 3: A method for scoring the model based on a set of metrics defined according to the parameters that are of concern to such an observation task.

Multi-agent robotic techniques and probabilistic modeling techniques are applied to minimize the initial acquisition time for first observing such ablating targets. The results of these techniques are evaluated using a set of metrics developed for both this modeling technique and for any approach to this observation problem.

The remainder of this thesis is organized as follows:

Chapter 2: The background and the motivation for this research. Related research is reviewed and compared to the solutions outlined in the thesis.

Chapter 3: The theory behind the methodology used for modeling ablating sources and the definitions of the metrics used to score the resulting model.

Chapter 4: The theory behind the methodology and algorithms used for assigning agents to particular search regions derived from the model.

Chapter 5: Setup and implementation of the algorithms used for controlling the robot, including the simulation and hardware environments used for the experiments.

Chapter 6: Experimental results which explore and validate the theory in Chapters 3 and 4.

Chapter 7: Concluding remarks and recommendations for future work.

CHAPTER 2

BACKGROUND AND MOTIVATION

The process of monitoring icebergs is related to other aquatic robot applications, particularly applications in which a set of *in-situ* environmental parameters must be monitored. While the aim of iceberg monitoring is, at its core, a target observation and tracking problem, other aquatic monitoring applications may not be concerned with physically moving targets. However, some of the techniques used are universal across the problems that these systems are intended to solve. In this section, some of the pertinent literature for existing iceberg and sea-ice surveys will be outlined, some of the previous work in aquatic monitoring applications will be examined, and other work that has similar properties to the iceberg tracking problem will be examined.

2.1 Previous Iceberg Surveys

As previously stated in Chapter 1, one of the most well-known and publically available collections of iceberg survey data comes from the International Ice Patrol (IIP) [2]. This organization was founded in 1914 as a result of the HMS *Titanic* disaster. The IIP monitors icebergs and other floating ice in the regions near the Arctic and the northern Atlantic Ocean. Any sea-based operation close to or within the "iceberg limit," an imaginary boundary line determined by the IIP that indicates the safest region with respect to the density of icebergs, uses data from this organization to maintain safety. Reports that indicate the current iceberg limit are issued daily, and they are possibly reissued depending on whether icebergs have been detected outside of the iceberg limit. One of the daily charts that the IIP generates is shown in Figure 2. The solid line represents the current iceberg limit, while the dotted line indicates the "sea-ice limit," which represents the extent of at least 1/10 of sea ice coverage at the time the chart was generated.

The data contained in the IIP databases is obtained through radar sweeps, visual observation (from both the IIP and outside observers), and satellite data. However, cases may exist such that an operation must be conducted from within the iceberg limit. In such a situation, the daily reports from the IIP would need to be supplemented with real-time situational awareness. To maintain situational awareness would require continuous visual monitoring by crew or use of ice-detecting radars. Such a task would be suited for an autonomous robotic observation system.

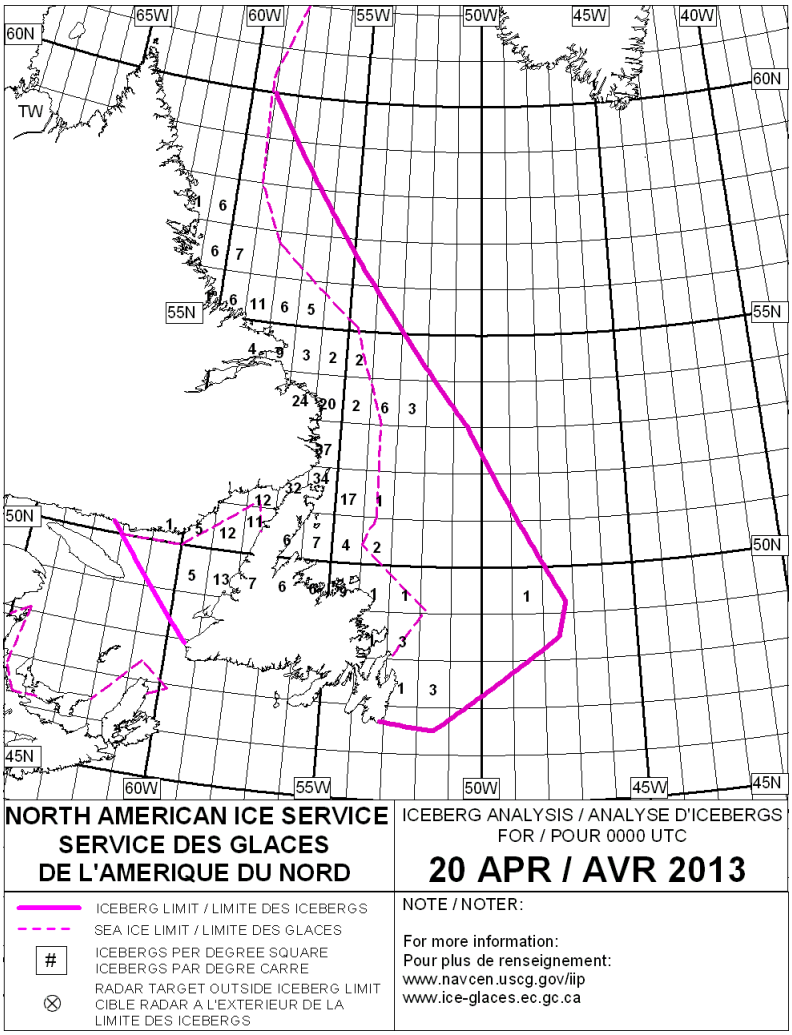


Figure 2. International Ice Patrol iceberg limit chart.

An extended study of the Austfonna ice cap in northeastern Svalbard, an unincorporated

region of Norway in the Arctic Ocean, also has been conducted [6–9]. While primarily concerned with the overall glacier dynamics in terms of mass loss, some of the common means by which icebergs are detected and tracked are demonstrated in this study. For example, in [6], synthetic aperture radar (SAR) imaging, satellite imaging, and airborne ice-penetrating radar are used to determine properties of ice motion and thickness. This study also provides useful data from which calving models can be derived, based on the rates of mass loss from the glacier.

Finally, many studies and observations have been conducted of the floating ice activity surrounding Antarctica, *e.g.* the study documented in [10]. One particular study focused on iceberg drift in the Weddell Sea [11]. The study used GPS buoys attached to icebergs to track their locations over time, making measurements of various iceberg properties prior to tracking and using the GPS data to observe iceberg drift and melt patterns within the Weddell Gyre, one of the ocean currents that flows around the Antarctic. This study demonstrates developing models and making measurements of iceberg movement through passive observation. Each iceberg that was tracked had its dimensions recorded; its initial recorded GPS position was recorded as well. Observations of iceberg drift velocity were derived from the iceberg position observations. In addition, the median volumes, iceberg drafts, and transit times were recorded, which provides a clear picture of iceberg behavior in the Weddell Sea.

This study also provided a complete data set of the icebergs that were monitored; this is additional data that can be used in model development. This method of tracking icebergs requires a more active component, since the buoys would have to be set up. For operations that simply require monitoring for danger, such an effort would only be necessary for the largest icebergs. Notably, results of these studies are usually incorporated into algorithms for forecasting sea ice activity, and, more recently, for producing what is known as a "nowcast" for sea ice [12]. A nowcast is an extremely short-term forecast. The forecasts themselves are produced by data assimilation [13] of offline data produced from other

surveys [14–16].

2.2 Robotic Over-Water Monitoring

Environmental monitoring of oceans and bodies of fresh water is a research activity that has some of the same characteristics as iceberg monitoring and is performed in many of the same ways: for example, one can set a buoy filled with sensing equipment out on the ocean, collect data, and then retrieve it later. However, depending on the region in which the monitoring is to take place, it is also possible to use mobile robots instead, in particular unmanned boats. These boats have the ability to travel to places that humans cannot, and can be teleoperated as opposed to autonomous, which is ideal for positioning the boats prior to data collection. As a result of the large areas that they need to cover, GPS is typically used for localization as opposed to GPS-free localization algorithms. In the case of fixed buoys, their locations are static; therefore, localization concerns are related to the uncertainty about the buoy location as a result of ocean currents. This section will outline a few examples of over-water monitoring, including buoys, underwater sensor platforms, and robot boats.

2.2.1 Buoy-like Data Recorders

Buoys can be considered the simplest method of collecting data over the ocean, similar to laying out a set of data loggers at key points as in a wireless-sensor network. However, often it is desirable to be able to observe certain properties of a particular water column that cannot be ascertained from a simple surface deployment; an autonomous instrument platform is useful for such an exercise. An older example of this type of platform is the Bottom Stationing Ocean Profiler (BSOP), detailed in [17]. Technically, the BSOP is not a robot: it is better described as an autonomous mobile sensor. The BSOP's ability to move vertically through a water column while drifting allows for the mapping of the water density field in addition to collecting other useful information about the water column. This information includes ocean current flow, temperature gradients, and optical characteristics.

The density field is the most important of these data items, as the field is generally difficult to measure without a human in the loop. And in the case of inclement weather on the ocean, deploying a manned ship to take these measurements is undesirable.

The platform has the appearance of a watertight cylinder. It contains a flotation apparatus that is deployed at key times to raise the platform to the surface to transmit its data; the BSOP continues to collect information about the water column as it ascends. After the platform has completed transmitting this data, it descends back to the ocean floor, collecting data as it moves downward. The BSOP stops moving once it strikes the ocean floor. This data may then be used to create a picture of the ocean properties of the water column at the location of the platform, as it also acquires GPS position and transmits that along with the data. Communications are via satellite, using specially formatted emails. Emails sent to the BSOP alter its behavior, while emails sent from the BSOP indicate the status of the platform and contain transmitted data.

2.2.2 Autonomous Boats

In addition to monitoring various properties of the ocean, the monitoring of properties of surface water (*i.e.*, fresh water) is also critical. The primary reason is that most of the population relies on surface water for potable drinking water, as do animals. Therefore, an autonomous and mobile system of monitoring rivers and lakes that are primary water sources would be ideal: as one cannot be certain where contamination can occur, placing buoys will not guarantee maximum coverage. The Robot Sensor Boats (RSB) in [18] are an attempt to solve this problem. They are autonomous, are small enough to reach areas that larger research vessels cannot, and are mobile, which allows them to provide sufficient coverage. Also, one person can monitor several of these boats simultaneously and in real-time, which is considerably safer than being near a source of contamination, if one exists. One other advantage is the simple fact that they are not fixed: if a body of water is only monitored using static buoys, criminal dumpers can introduce contaminants in areas that do not have sufficient coverage by said buoys and thus not be detected until it is too late.

Such autonomous boats would not be viable if they were not relatively inexpensive in the same sense that wireless-sensor networks are; therefore, the RSBs are constructed from commercially-available materials. The boats are small kayaks with ducted propulsion that allows for differential steering and is loaded with standard measuring and communication equipment that is all connected to the boats' computers. Communication is accomplished via the use of a cellular modem, and the data is aggregated on a remote server. The robot boats themselves are safe for people in the waters and allow operation for over six hours. Finally, there is one last thing to consider, and that is efficient deployment of the boats. They were kept within 100 lbs to allow two people to easily launch one.

The key aspect of this system is the way that the boats' control and their data are tied together, via the authors' Multilevel Autonomy Robot Telesupervision Architecture (MARTA) [19]. The system allows for complete autonomy and for allowing humans to take teleoperated control of the robot boats as necessary. Goals such as directing a group of boats to check a certain region of water can be assigned, and the system will automatically break down the task into chunks, and assign each sub-goal to one of the assets involved in the investigation. The activities of the robots are then monitored by the "Robot Team Coordinator," one of the system components and by a human operator. The human operator always has the option of intervening directly and performing a manual replan of a boat's route or taking direct control using teleoperation.

The planned routes themselves allow for a degree of error, since it is not imperative that the boat reaches each waypoint of its route precisely. Each waypoint has an "achievement circle" which allows for fuzzy waypoint navigation as a function of both the steering capability of the boats and the accuracy of the onboard GPS units, augmented with onboard compasses. This is intended to mitigate the positional uncertainty in using GPS navigation.

The authors' initial experiments with the sensor boats in a man-made lake near Carnegie-Mellon University allowed quick determination of several, near complete mappings of the lake, in terms of dissolved oxygen and pH among other quantities. Such a task could not

have been easily done with stationary sensors. The ultimate goal is to be able to perform *in-situ* scanning of an area without significant initial preparation.

Aside from freshwater properties that can be measured from the surface, as the RSBs do, there is always a need to push down further into a water column and gain more information. This next autonomous boat, referred to as an Autonomous Surface Vehicle or ASV, has a sensing apparatus that allows for such a function [20]. Unlike the RSBs, this vehicle is fully autonomous without the option for teleoperation: the purpose of the ASV is to make circuits of a defined route composed of waypoints in a body of water, generate measurements of its current water column using its onboard sensor suite, and communicate this information back to shore. This data can then be used for early warning of certain events, such as algal blooms. The sensor suite is composed of an optical methane detector, a sonde for general water measurements, a wind sensor, and profiling sonar.

For navigation, the ASVs are equipped with a GPS, digital compass, depth sensor, laser scanner, and camera. With the position information, directing an ASV toward its goal point is accomplished through the technique of virtual forces, an adaptation of artificial potential fields [21]. In summary, a goal force attracts the ASV, an alignment force attempts to keep the vehicle on a straight line between the start and goal, and a repelling force pushes the vehicle away from any obstacles, which may be slow-moving or stationary. A set of feedback controllers then set the motor commands. The end result is the ASV essentially pulling itself from waypoint to waypoint.

Finally, to assuage power concerns, the robot boats use solar panels to recharge their batteries. The ASV has the ability to interface with the floating wireless-sensor network on the lake, which the authors configured as part of their experiments. This sensor network acts as the primary method by which the operators and the ASV communicate; the ASV sends its measurements via this sensor network to shore. This allows for decentralized communication that does not require line of sight.

2.3 Robotic Harmful Algal Bloom Monitoring

Closely related to over-water monitoring is the monitoring of harmful algal blooms. The methods that are used in robotic systems that are directed to monitor harmful algal blooms is similar to methods one would use to monitor icebergs; however, the primary difference is that only one or two large "targets" exist for the case of an algal bloom. Therefore, the goal in this case is to determine the extents and spread of a single bloom, rather than acquire and track individual targets.

The ASV in [20] has as one of its alternate functions the ability to monitor the water for the parameters necessary for such blooms to form. However, there are certain cases when monitoring only for changing water parameters might not be sufficient; there may be other and perfectly natural reasons for certain parameters to be out of balance. The work of a few groups of authors will be primarily discussed here. One may note that a significant amount of this work has been concerned with harmful algal blooms in the Pacific Ocean.

A particular series of papers [22–24] describe research efforts to extend the Robot Sensor Boats of [18] for this sort of application through the use of a new system that extends the MARTA architecture [19]. This new system is called the Telesupervised Adaptive Ocean Sensor Fleet (TAOSF), which is composed of five separate components: the OASIS ASV System, the Multi-Platform Simulation Environment, the Platform Communicator, the Adaptive Sensor Fleet (ASF), and the System Supervision Architecture (SSA). In terms of assigning goals and directing the robot boats, the SSA is the component that is directly based on the MARTA infrastructure; in terms of algorithms, it is very similar to this earlier work. Some of the main differences are the use of more sophisticated ASVs, with more sensors as opposed to the RSBs made from off-the-shelf components and the additional control systems provided by the ASF. The significant aspect of this system is how it is applied to the task of harmful algal bloom monitoring.

The desire to detect and monitor harmful algal blooms has increased in recent years as a result of the detrimental effects on humans and sea life. Time of year, salinity, and

sea-surface temperature are metrics used to predict abundance of certain types of algae; in this paper, it is the dinoflagellate *Karlodinium micrum*, notable for its red color. The sensor suite on the OASIS ASVs has the ability to measure water salinity and conductivity, the temperature of the surface, and chlorophyll; all of which, collectively, can be used to analyze and predict algal bloom formations. To test the system, rhodamine dye was used to simulate the algal bloom; the diffusion of the dye was then mapped aerially using an aerial observation platform integrated with the TAOSF as part of characterizing the performance of the system. To correlate the observed time and space varying properties, the Inference Grid (IG) model [25] was used as a general spatial representation. An IG is a Markov random lattice that stores the collected sensor information, which is further used in inferencing and mission planning. Mapping the dye patch was accomplished via an unsupervised clustering algorithm; the clusters were identified by the known spectral characteristics of the dye. A similar technique can be used for the algae itself. The ASF system can then plan and execute a path for the ASVs to follow based on known initial observations; in the case of characterizing the system, a spiral path was used to observe the simulated algal bloom. The ASVs can then form a probabilistic map of where algae has been found, and then subsequently this data can be analyzed and a so-called "blob" hypothesis can be formed. These hypotheses are a method of aggregating the values of the probabilistic map to generate an approximate "terrain map" of the algal bloom density, taking the uncertainties in the spread of the bloom into account.

2.4 Search-and-Rescue Applications

Search-and-rescue operations, like tracking icebergs, also require minimization of the time to acquire a target. Consider searching for survivors in a collapsed building as the result of a natural disaster using a robot expressly designed for that purpose. Such robots often have dedicated operators that use teleoperation to direct the actions of the robot, and a limited amount of autonomy on the part of the robot, so that the human operator can spend more

time concentrating on the rescue effort. The robot typically integrates various sensors and uses sensor fusion techniques to better assist the searcher in finding targets.

An examination of various user interfaces is provided by the work summarized in [26], which gives the results of several approaches to the problem in a simulated urban search and rescue (USAR) environment during the Robocup 2003 USAR competition and provides recommendations for sensory interfaces for future robotic platforms intended to work in this area. A summary of key issues in this area is given by [27], which includes the dangers that are involved, and an approach to the problem using "micro-bots" that are assigned to various disaster regions such as a collapsed building as the result of an earthquake. Additional applications of artificial intelligence techniques to the problem are also examined, such as neural networks and expert systems.

Another application of multiple robots to USAR as opposed to a single operator and robot to maximize coverage is given in [28]. A simulation was developed using the Unreal engine to determine any sensory and human-robot interaction (HRI) issues that may arise, and from the results of that simulation, developed robotic platforms that would be useful for USAR tasks. A probabilistic model was also used to determine whether locations contained victims: each probability of victim detection was weighted by a confidence value assigned to the corresponding sensor.

In [29], an "ant-like" approach is shown that provides for maximum unpredictability of the resulting robot paths by using a frequency distribution to determine the locations visited, which would improve area coverage. Unpredictable paths, however, may result in paths that miss a wide swath of targets, which does not contribute to the stated goal of minimization of initial target acquisition time.

2.5 Multi-robot Observation of Multiple Moving Targets

In terms of existing robot observation problems, the one that relates most closely to the iceberg observation problem is Cooperative Multi-robot Observation of Multiple Moving

Targets (CMOMMT) [30]. The primary objective of solutions to this problem is the determination of the best placement of robots that will observe targets that move through a region of interest. This problem class can be adapted through changes to the base assumptions, but the original problem is difficult enough that many solutions to it have been investigated by researchers; indeed, it has been proven to be NP-hard. Some of the solutions to the CMOMMT problem will be examined in this section.

One of the first solutions to the CMOMMT problem used a behavior-based approach [30] [31]. A set of heuristics for placing the robots at the correct points was developed, and force vectors were used for navigation. The force vectors were designed such that a robot was attracted to a target and repulsed from other robots. To deal with the issue of local minima, motivational heuristics were used to weight the contributions of each target's force field. The result is that all targets are efficiently observed by all robots. This solution is referred to as *A-CMOMMT*. Follow-up research used a learning algorithm to learn the correct behaviors for observing a particular target space, as opposed to hand generating the behaviors as in A-CMOMMT [32].

This research was concerned with indoor applications; soon solutions appeared for the outdoor problem. Issues such as environment occlusion occur in these sorts of environments; one of the solutions used a region-based method for solving the CMOMMT problem [33]. This approach divided the environment into convex regions based on landmarks, and the robots attempt to maximize the number of targets that are observed within a region. The regions are used to determine target and robot density: the main idea is that if there are more targets than robots in a given region, then another robot should move to that region to assist the other robots. Otherwise, any otherwise untasked robots should explore unoccupied regions for targets. This research was followed up by a generalized approach [34] that eliminated the need for landmarks to determine search regions. The regions, instead, were computed as a function of a robot's sensing capability. This method allowed for a number of advantages, such as automatic grouping of targets and a removal of the need for motion

planning. However, the function used to determine target region "urgency" is unbounded; this problem was resolved by limiting the size of the search area.

Another application for this problem was a "tunably decentralized" approach [35]. In this approach, all of the agents knew the locations of all of the other agents and the targets. The k-means clustering and hill-climbing algorithms were then applied separately and jointly to determine observation points for the agents. The algorithms were tunable in that the number of subsets generated changed based on a single parameter; *i.e.*, the k clusters in k-means and the number of observer positions for hill-climbing. In their study, the authors find that k-means is unaffected by decentralization; they conclude this result occurred as a result of the very nature of k-means clustering. That is, targets far away from a particular cluster will never be claimed by that cluster. As for comparing the two algorithms, hill-climbing worked best for slow-moving targets, and k-means worked best for fast-moving targets. Combining the two produced a good "middle ground," according to the authors.

In [36], the authors built upon A-CMOMMT to develop *B-CMOMMT*, which is a *behavioral* version of A-CMOMMT. Essentially, they treated the problem within a decision-process framework. They use the same force-vector paradigm as in A-CMOMMT; the main difference is that they used hard-switching among behaviors to direct the robots, as opposed to the softer heuristic approach in A-CMOMMT. These behaviors included a follow target mode, help mode (a robot may assist others in monitoring targets), and an exploration mode. They demonstrated superior performance to A-CMOMMT with this algorithm. The authors further refined their results in [37], adding improved target loss prediction and modifications to the calls for help. They additionally proved that their algorithm is stable, and they demonstrated greater performance with their improvements to the algorithm.

Finally, a decentralized, model-predictive approach to the CMOMMT problem is taken by the authors of [38]. The primary focus was ensuring that their approach had a certain

degree of optimality. Using the mixed-logical-dynamical (MLD) framework, they developed a model for the agents and a cost function for the task, with which they could solve the resulting optimal control problem. Their decentralized approach allowed for the size of the subproblem to be solved to vary (*e.g.*, individual target and agent counts), and the approach did not require constant communication with a central controller. Their results demonstrated that their algorithm is superior to the A-CMOMMT approach.

2.6 Summary of Related Work

Previous work related to the iceberg observation problem can be summarized by considering the following categories, and the pieces that may be taken from each of these categories to improve *in-situ* observation of icebergs for situational awareness:

- *Iceberg surveys using aircraft and radar imaging systems*: Such surveys provide a bigger picture for situational awareness. The data provided can be used as *a-priori* planning data for missions for efficient iceberg observation.

- *Over-water monitoring using buoys and autonomous vehicles, including monitoring of harmful algal blooms*: The techniques used in these observation methods, such as clustering and coverage patterns, can be adapted and improved for the iceberg problem.

- *Search-and-rescue applications and monitoring of moving targets*: Search-and-rescue applications are concerned often with coverage to determine the most efficient ways to find survivors, which may be moving targets. Techniques used to improve coverage can also be applied to iceberg observation. The latter provides a starting point for a model for the iceberg observation problem, and solutions to the problem of monitoring moving targets can be examined within the context of the ways that the iceberg observation problem differs.

In any case, the techniques and algorithms developed by previous researchers can be viewed as inspiration for developing efficient algorithms for observing icebergs. The particular interest, as previously stated, is for *in-situ* observation, which is not an area of primary concern for most iceberg monitoring operations, which are more interested in producing forecasts and emphasizing that operations stay outside of regions of moderate to high iceberg risk. However, operations that must be conducted within such regions provide a motivation for developing a robotic system that can efficiently locate potential iceberg threats.

CHAPTER 3

METHODOLOGY FOR MODELING ABLATING SOURCES

In this chapter, the methodology by which ablating target sources are modeled is outlined. The problem definition is motivated by an existing robotic observation problem. This original problem is extended by a set of additional assumptions, to make it fully applicable to the ablating target source observation problem. A methodology is then defined that uses probabilistic techniques to develop a model for determining the regions at which observation resources should be focused. From this model, a set of metrics are defined that extract useful information from the model that allow for evaluation of algorithms that are used to address the observation problem after a model has been generated, in addition to providing quantities that can be used in cost functions that provide for allocation of observation resources.

3.1 Problem Definition and Assumptions

3.1.1 Overview

Since masses of floating ice are moving targets, and an observation region can be defined in which ice masses float into and out of the region, the problem of remotely observing these targets is very similar to the class of well-known, robotic observation problems defined as Cooperative Multi-robot Observation of Multiple Moving Targets (CMOMMT) [30]. The primary objective of solutions to this problem is the determination of the best placement of robots that will observe targets that move through the region of interest. This placement must maximize the amount of time that any given target is observed.

This problem class can be adapted through changes to the base assumptions, as described below, to fit many observation tasks; however, the general problem itself is difficult enough that many solutions have been developed by researchers [30–38]. The CMOMMT problem is useful as a framework that can be extended to obtain a general model for the iceberg observation problem.

3.1.2 Definition

The standard CMOMMT problem is defined using the following definitions:

$\mathcal{S}$: A two-dimensional, polyhedral region.

$\mathcal{R}$: A set of M robot positions r_i.

$O(t)$: A set of target positions $o_j(t)$, such that $o_j(t)$ is contained within $\mathcal{S}$ at time t. The cardinality of $O(t)$ at time t is $N(t)$.

The robots have omnidirectional sensors that are limited in observation range. The target positions contained in $O(t)$ can change, and the initial positions are not known in advance. The members of $O(t)$ are assumed to enter and exit $\mathcal{S}$ through well-defined entrances on the boundary of $\mathcal{S}$. Figure 3 is an illustration of an example layout for this problem. The thick bars at the top and bottom of $\mathcal{S}$ are target entrances and exits.

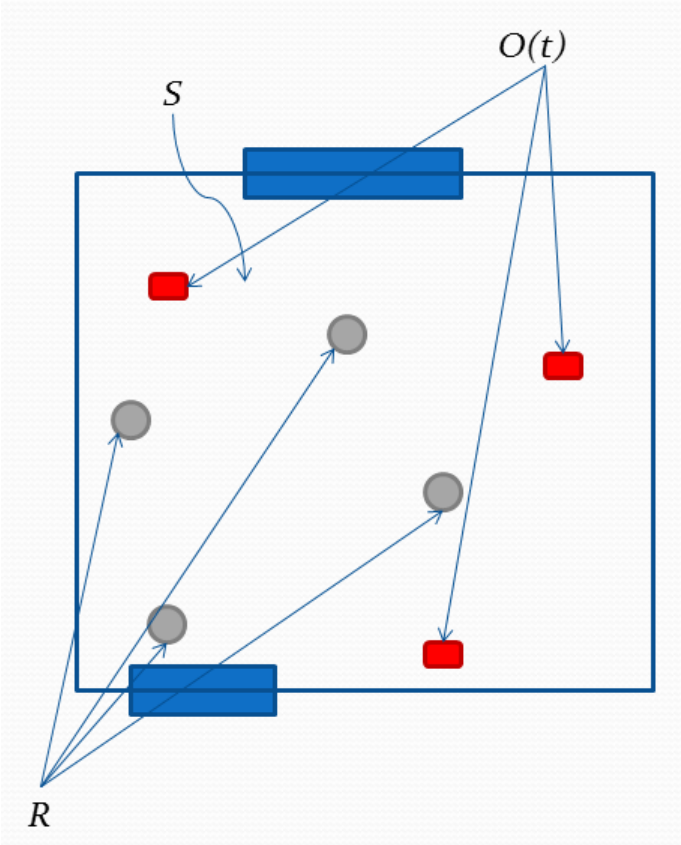


Figure 3. An example layout of the objects defined in Section 3.1.2.

With these definitions and assumptions, for CMOMMT, the $M \times N(t)$ matrix $A_{ij}(t)$ for a specific time t can be defined with the elements

$$a_{ij}(t) = \begin{cases} 1 & \text{if robot } r_i \text{ is monitoring target } o_j(t) \text{ in } \mathcal{S} \\ 0 & \text{otherwise.} \end{cases}$$

With this matrix, the CMOMMT objective function for time t, with a range 0 to T, divided into discrete Δt steps, is defined to be[1]

$$\sum_{t=0}^{T} \sum_{j=1}^{N(t)} \bigvee_{i=1}^{M} a_{ij}(t). \tag{1}$$

The goal of the CMOMMT problem is to maximize this objective function: the time that each object is being monitored by at least one robot under some set of assumptions. Additional base assumptions exist for the CMOMMT problem; however, they are not relevant to the framework as given here. Further details are contained in the reference [30].

For the problem of ice mass generation and observation, the targets are assumed to have a highly variable but bounded velocity, which is reasonable to assume for an object under the influence of ocean currents. An object called a *target source* will also be defined: a target source is any member of $O(t)$ at $t = 0$.

To account for the ice mass generation and observation of environmental phenomena, the CMOMMT assumptions are modified as follows:

1. No specific target entrances or exits are defined at the boundary of $\mathcal{S}$. Targets may exit from any location on the boundary of $\mathcal{S}$. This assumption accounts for iceberg behavior in the middle of the ocean; no boundaries prevent iceberg drift.

2. Targets are generated from target sources located within the interior of $\mathcal{S}$; *i.e.*, the set $O(t)$ grows in membership over time as the target sources undergo ablation.

3. *A-priori* information regarding approximate target source locations is available at

[1]The $\vee$ operator indicates a logical "or" over all members in the set.

$t = 0$. For example, information from the IIP sightings databases may be used to determine these target source locations.

Assumptions 1 and 2 are derived directly from the ice mass problem as discussed in Chapter 1. Note that the fact that the targets are generated from ablation suggests that an upper bound on the number of targets that a given source can generate in a time interval exists. That is, a target source will eventually break up and cease to exist as a unique entity in $O(t)$. Assumption 3 is necessary to provide an initial allocation of the agents directed to observe the target sources.

The objective of this modified problem is slightly different from the general CMOMMT problem. Since early detection is paramount, the amount of time required to obtain the *initial contact* on a target contained within $\mathcal{S}$ must be minimized. The contact time, denoted by T_s, is defined as the time at which a robot's sensors detect the target referenced from the start of the ablation process. These target contacts can be obtained by a variety of different search algorithms. For the purposes of this problem definition, the target contacts will be obtained by the agents using some type of search pattern [39] [40] [41] to execute their search, and any further target tracking will be accomplished by agents dedicated to that task. The search patterns used in this research are discussed quantitatively in this chapter; Chapter 5 provides a further qualitative discussion of the search patterns as they are incorporated in a robot controller.

In addition, as a result of sensor uncertainties, the sensors used by the robots will not have a probability of detection equal to 1. Hence, the contact time T_s for a *single target* is the contact time of a "perfect" sensor plus zero-mean noise. The noise variance is dependent on the sensor errors. However, T_s is different for each of the targets contained in $O(t)$ for a given t as a result of the different robot positions r_i with respect to the targets $o_j(t)$. This dependency means that to obtain the expected value of the contact time for a given target source, the expectation must be taken over all of the targets.

Given these conditions, the objective function for the modified CMOMMT problem can be defined to be the following:

$$\min_{\mathcal{R}} E[T_s|\mathcal{O}(t)], \tag{2}$$

where $E[\cdot]$ denotes the expected value operator.

3.2 Summary of Methodology

With the problem defined in Section 3.1, a methodology can be outlined for minimizing the objective function using multiple robotic agents. For areas of higher activity on a target source, more sensing resources will be required to ensure that all targets are observed in a minimum amount of time. Therefore, more resources should be allocated to regions with a high probability of a new target being generated. As the target probability for a particular region decreases, agents should be reallocated in a more equal manner around a target source. This agent allocation scheme will not only minimize the initial contact time, but it will also reduce the uncertainty in quantifying the activity around a particular target source.

To determine the appropriate probability density of the targets, observations must be incorporated into a model. Observations are composed of the following elements: the position at which the target was first observed, the time at which the target was first observed (observation time), and which agent made the observation. The model incorporates these observations and the current number of observed targets across all agents and the *a-priori* probability density of a new target being formed, which is obtained from a previous iteration of this process. In the case of the very first observations being made, the *a-priori* density is initialized using the *a-priori* observed position of the target source.

On-line analysis of the *a-posteriori* density will indicate whether to undergo a reallocation of resources. When no *a-posteriori* density is avaliable, *e.g.*, during mission start, a default allocation of resources will be used. This default allocation is the following: the region of interest is divided into M equally-sized cells, with each agent placed at the center of each cell. A metric of variability will be developed to indicate when the allocation

should change, and new search regions will be devised from the most active regions in the model. The level of activity will be computed by determining which regions in the model have the highest probability of generating a new target.

An example of such a model is shown in Figure 4. The glacier-sea interface is indicated by the jagged line on the left. There are two regions of high ice mass probability; therefore, a reallocation of resources would either split the number of currently active agents between the two regions, or allocate more agents to the mode of the distribution with a greater spread (corresponding directly to a greater area of required coverage). Note that the probability of a calving event is not necessarily proportional to the size of the coverage area; glacier properties such as flow rate have a greater influence on the probability.

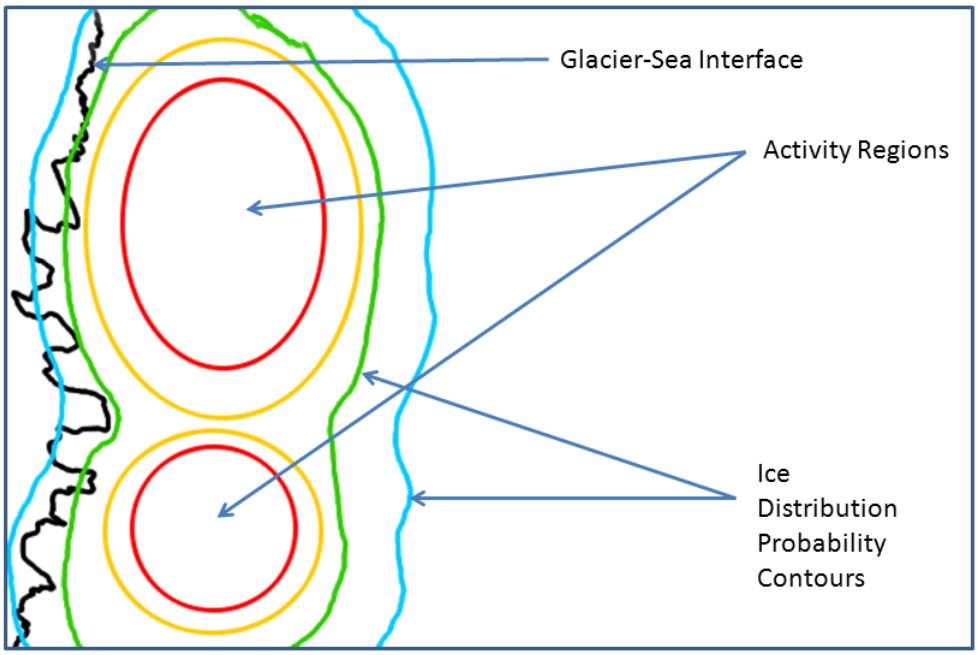


Figure 4. Example probabilistic model of an ablating source.

Note that certain aspects of this methodology have some similarities to the coverage problem. However, in the coverage problem, the region of interest is generally static and the problem of interest is determining the most efficient means of observing the entire region or some aspect of the entire region. What is outlined here resembles an *adaptive* coverage solution, since a model of the target sources is determined and the resources (*i.e.*,

agents) and regions of coverage are adapted to fit that model.

3.3 Definitions and Assumptions Specific to Iceberg Observation

Given the methodology in the previous section for probabilistically modeling iceberg activity regions for reallocating sensor resources and the problem definition in Section 3.1.2, additional definitions and assumptions are required to further constrain the problem with respect to the methodology and provide a framework in which algorithms can be developed. In this section, these additional definitions and assumptions are provided.

3.3.1 Definitions

The following definitions further define the overall problem. To begin, $\mathcal{S}$ from Section 3.1.2 is more precisely defined, as are the iceberg ablation regions.

Definition 3.3.1.1 *$\mathcal{S} \subset \mathbb{R}^2$ is the rectangular and topologically connected region of interest.*

$\mathcal{S}$ defines the region of the ocean in contact with the glacier with the ablation points that are to be observed.

Definition 3.3.1.2 *U is the set of ablation points in $\mathcal{S}$;* i.e.*, $U \subset \mathcal{S}$. The cardinality of U is finite: $|U| = l$, $l \in \mathbb{N}$. Individual ablation points are $u_i \in U$, with $i = 1...l$.*

For each ablation point u_i, there is a corresponding random process p_i that results in the generation of targets from the ablation point u_i.

Definition 3.3.1.3 *The random process p_i, for $i = 1...l$ and associated with an ablation point u_i, is a homogeneous Poisson process with intensity λ_i: $p_i \sim$ Poisson (λ_i).*

The random process p_i for each u_i results in sets of iceberg target trajectories.

Definition 3.3.1.4 *$B_i(t) \subset \mathbb{R}^4$, for $i = 1...l$, is the set of all target trajectories at a specific time t resulting from the ablation processes of a specific u_i. $B_i(t)$ is referred to as a* target stream. *$N_i(t) \in \mathbb{N}$ is the cardinality of $B_i(t)$ at time t.*

Definition 3.3.1.5 *$\beta_{i,d}(t) \in B_i(t)$, with $i = 1...l$ and $d = 1...N_i(t)$, is an individual target trajectory state vector at time t, containing the position and velocity at that time.*

Figure 5 illustrates each of these definitions graphically, with respect to the iceberg observation scenario.

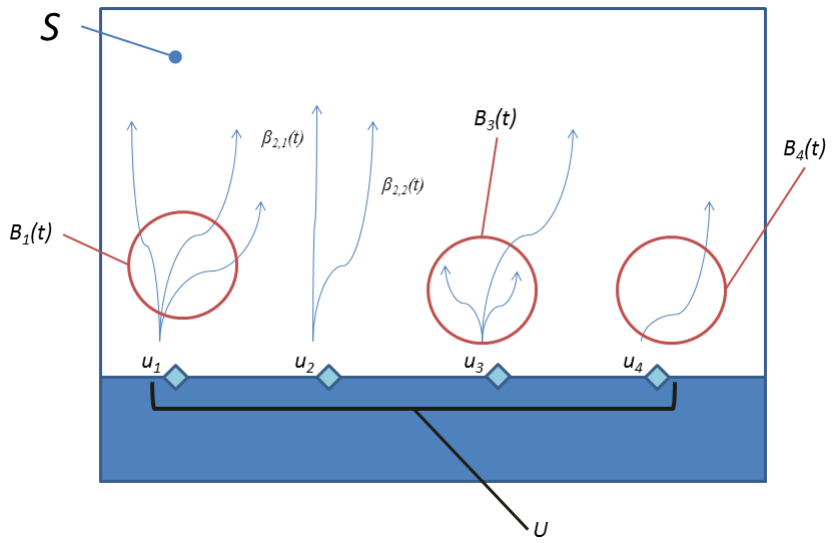


Figure 5. Diagram of the problem definitions.

Before defining the model and the mathematical objects associated with it, two function definitions are necessary to provide a convenient way of extracting state from a trajectory $\beta_{i,d}(t)$.

Definition 3.3.1.6 *The function $g : \mathbb{R}^4 \longrightarrow \mathbb{R}^2$ returns the velocity vector of a specific $\beta_{i,d}(t)$.*

Definition 3.3.1.7 *The function $h : \mathbb{R}^4 \longrightarrow \mathbb{R}^2$ returns the position vector of a specific $\beta_{i,d}(t)$.*

With the region of interest and the objects within it defined, the probabilistic model used to model the spread of icebergs and their probability of generation may be defined. The model parameters are defined according to the parameters of a Gaussian mixture distribution, which will be further elaborated upon in Section 3.4.1.

Definition 3.3.1.8 *$Q = \left\{q_j\right\}_{j=1}^{k}$ is the set of mixture components that represents the target trajectory dispersion within the region of interest $\mathcal{S}$. The cardinality of Q is finite: $|Q| = k$, $k \in \mathbb{N}$.*

Definition 3.3.1.9 *$Z_j \subset \mathbb{R}^2$ is a set of target measurements associated with a mixture component. The cardinality of Z_j is finite: $|Z_j| = n$, $n \in \mathbb{N}$.*

Definition 3.3.1.10 *Target measurements $z_{j,m} \in Z_j$, for $m = 1...n$, are noisy position measurements of iceberg position. That is, for some time t, $i \in [1, l]$, and $d \in [1, N_i(t)]$:*

$$z_{j,m} = h(\beta_{i,d}(t)) + w, \tag{3}$$

where $w \in \mathbb{R}^2$ is a sensor noise term.

Definition 3.3.1.11 *Each member $q_j \in Q$, with $j = 1...k$, is a mixture component. Each mixture component q_j is defined as the 4-tuple*

$$q_j = \left(\mu_j, \Sigma_j, Z_j, \bar{v}_j\right), \tag{4}$$

where $\mu_j \in \mathbb{R}^2$ is the component mean vector; $\Sigma_j \in \mathbb{R}^{2\times2}$ is the component covariance matrix; $Z_j \subset \mathbb{R}^2$ is the set of target measurements associated with the component; and $\bar{v}_j \in \mathbb{R}$ is the estimated velocity magnitude of targets within a component.

The following definitions further elaborate on the structure of the set of target measurements Z_j.

Definition 3.3.1.12 *For $\Omega(t) = \bigcup_{i=1}^{l} B_i(t)$, $W_j(t) \subset \Omega(t)$ is the set of unique target trajectories generating the target measurements $z_{j,m} \in Z_j$. The cardinality of $W_j(t)$ is finite: $|W_j(t)| = \alpha_j(t)$.*

Definition 3.3.1.13 *The number of unique target trajectories $N_T(t)$ acquired across all sensors at the time t is the following sum:*

$$N_T(t) = \sum_{j=1}^{k} \alpha_j(t) \tag{5}$$

Finally, for evaluating the acquisition time and other parameters that involve iceberg drift, a time window is required. All times are measured relative to the beginning of ablation activity. The following definition constrains the total time that bounds the number of target measurements acquired by the set of sensors.

Definition 3.3.1.14 *The time T is the time at which all possible targets ablated from all sources in U have exited the region of interest $\mathcal{S}$.*

3.3.2 Assumptions

Based on the problem definitions of the previous sections, the following assumptions further constrain the overall iceberg observation problem.

1. Once created, targets move at a constant velocity with respect to their originating ablating source; that is, the targets move according to laminar flow. Precisely, $g(\beta_{i,d}(t)) = V$, $V \in \mathbb{R}^2$ is a constant vector.

2. Within the region of interest $\mathcal{S}$, target streams $B_i(t)$ do not cross. That is, $B_a(t) \cap B_b(t) = \emptyset$ where $B_a(t)$ and $B_b(t)$ are target streams with indices $a, b \in [1, l]$ at time t.

3. The target distribution in $\mathcal{S}$ is not a uniform distribution over $\mathcal{S}$.

4. One retrieved target measurement $z_{j,m}$, with $j \in [1, k]$ and $m \in [1, n]$, corresponds to one target trajectory $\beta_{i,d}(t)$, with $i \in [1, l]$ and $d \in [1, N_i(t)]$.

5. The sensor noise term w is drawn according to a zero-mean, Gaussian white noise process with noise covariance matrix $\Sigma_w \in \mathbb{R}^{2\times2}$. That is, $w \sim \mathcal{N}(0, \Sigma_w)$.

6. Target measurement false alarms $z_{fa} \in \mathbb{R}^2$, $z_{fa} \in Z_j$ are uniformly distributed within $\mathcal{S}$ with a constant probability P_{fa} of occurring.

7. A target measurement $z_{j,m}$ is either a false alarm z_{fa} or a true target measurement, but not both.

8. More than one component q_j of a model Q can contain a measurement of the same target.

9. Estimated target velocity $\bar{v}_j$ is constant within a model component q_j. (*i.e.*, $\bar{v}_j = v$ where $v \in \mathbb{R}$ is a constant).

10. The probability of detection P_D of targets generated from all members of U over $\mathcal{S}$ is constant over the time interval $0 \leq t \leq T$.

3.4 Modeling Ablating Sources

The methods by which these ablating sources can be modeled will be discussed in the following sections. The results of these methods are twofold: a model which can be used to predict future iceberg movement based on the regions of activity and efficiently search for new targets, and a set of metrics that can be used to characterize the behavior of the icebergs which will assist in assigning resources and predicting future iceberg behavior.

3.4.1 Model Definition and Computation

A probabilistic model for the target source as described in Section 3.2 and defined in Section 3.3.1 as Q would allow for multiple modes and for quantifying the spread of the targets that have been generated. With respect to the given requirements, a Gaussian mixture model (GMM) [42] is one appropriate way of modeling the regions of activity on a target source; the individual components in the mixture will be assumed to directly correspond to the regions of activity at the glacier-sea interface. Hence, the mixture will act as the *a-posteriori* density described in Section 3.2; the *a-priori* density is derived from the initial parameters for the search conducted by the agents, which are set as part of a mission planning process. Each agent will maintain its own "picture" of the model. This allows for each agent to make its own decisions based on how it views the model and redundancy in the case that an agent is disabled.

GMMs are defined in terms the means and covariances of their components; *i.e.*, the

probability density function $p(x)$ for a GMM is defined as the weighted sum of mixture components

$$p(x) = \sum_{j=1}^{k} w_j \phi(x|\mu_j, \Sigma_j), \tag{6}$$

where each w_j is the component weight, x is the vector-valued parameter at which to evaluate the function, and μ_j and Σ_j are the component means and covariance matrices, respectively. ϕ is the multivariate Gaussian probability density function

$$\phi(x) = (2\pi)^{\frac{\kappa}{2}} |\Sigma_j|^{\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)}, \tag{7}$$

where κ is the dimension of μ_j. The weights of each of the components correspond to the probability that a measurement belongs to a certain component.

As previously stated in Section 3.3.1, each component in the mixture will correspond to a region of ablation activity. The covariance matrix of each component will be used to derive the extents of a region that the agents must monitor. Since the component weights are a measure of the influence of a component, they will be used in ranking which component requires more observation resources in concert with the spread of the observation.

To generate the model, the agents must obtain *measurements* of target state. Before defining what a measurement contains, each target's state vector $\mathbf{x}$, a single point of the target trajectory $\beta_{i,d}(t)$ at a given time t, is defined to have the following form:

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}, \tag{8}$$

where (x, y) are the target position components, extractable using the function $h(\mathbf{x})$, and $(\dot{x}, \dot{y})$ are the target velocity components, extractable using the function $g(\mathbf{x})$.

Measurements of the target states must be shareable across agents, to provide for the generation of a complete picture of the target source model at each agent. Therefore, each measurement will be labeled with an *agent identifier*, *target identifier*, and the *time* at which

the measurement was obtained. The remainder of the measurement vector consists of position observations, perturbed by noise. A complete measurement vector $\mathbf{z}$ will be defined as

$$\mathbf{z} = \begin{bmatrix} \Lambda \\ \eta \\ t \\ x_m \\ y_m \end{bmatrix}, \tag{9}$$

where $\Lambda \in \mathbb{Z}$ is the label assigned to the target, $\eta \in \mathbb{Z}$ is the agent identifier, t is the observation time, and (x_m, y_m) is the noisy position observation, according to the assumptions for a target measurement in Section 3.3.2. Note that target velocity is not part of a measurement of the state. For the purposes of this model, the sensors are not assumed to have the ability to measure velocity (*e.g.*, Doppler measurements). This assumption is made because the target velocities of drifting icebergs are not very high in magnitude, and may be estimated to a certain degree of accuracy from sequential measurements of position.

Given a set of measurements of target position, the GMM components can be computed. *Expectation-maximization* [43] (EM) is used to obtain the GMM components once a sufficient number of measurements has been obtained. As measurements are collected, either through the actions of an individual agent or by obtaining measurements from other agents, the EM algorithm is run again on the new set of measurements, developing a clearer picture of the regions of activity for a particular target source. In this case, the spread of the iceberg positions with respect to the target sources is of interest, so the (x_m, y_m) components of the measurement vector $\mathbf{z}$ are used as the samples in the EM algorithm.

The EM algorithm is initialized by first running *k-means clustering* [44] over the position measurements. The results of the clustering are used as the initial means in the EM algorithm, with the covariances initialized to identity matrices. A limit is also placed on the number of iterations to prevent infinite loops from occurring as a result of the algorithm

not converging to a suitable solution.

The version of the EM algorithm as given in [43] is then run over the measurements and the initial means and covariances with one difference: the log-likelihood is not scaled by the number of measurements n. Scaling the log-likelihood, for this particular application, causes numerical stability issues in the algorithm. That is, the resulting minuscule likelihoods prevents convergence to a solution prior to reaching the limit on the number of iterations. The modified algorithm is as follows, where m is the index of the current algorithm iteration:

1. *Initialization*: Initialize as previously described for the initial component estimates, and calculate the initial log-likelihood:
$$\ell^{(0)} = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{k} w_j^{(0)} \phi\left(z_i | \mu_j^{(0)}, \Sigma_j^{(0)}\right)\right).$$

2. *Expectation Step*: For $j = 1, ..., k$, compute the association probabilities as follows, with $i = 1, ..., n$:
$$\gamma_{ij}^{(m)} = \frac{w_j^{(m)} \phi\left(z_i | \mu_j^{(m)}, \Sigma_j^{(m)}\right)}{\sum_{s=1}^{k} w_s^{(m)} \phi\left(z_i | \mu_s^{(m)}, \Sigma_s^{(m)}\right)}$$
and
$$p_j^{(m)} = \sum_{i=1}^{n} \gamma_{ij}^{(m)}.$$

3. *Maximization Step*: For $j = 1, ..., k$, compute the updated component parameter estimates:
$$\begin{aligned}
w_j^{(m+1)} &= \frac{p_j^{(m)}}{n}; \\
\mu_j^{(m+1)} &= \frac{1}{p_j^{(m)}} \sum_{i=1}^{n} \gamma_{ij}^{(m)} z_i; \\
\Sigma_j^{(m+1)} &= \frac{1}{p_j^{(m)}} \sum_{i=1}^{n} \gamma_{ij}^{(m)} \left(z_i - \mu_j^{(m+1)}\right)\left(z_i - \mu_j^{(m+1)}\right)^T.
\end{aligned}$$

4. *Convergence*: Compute the log-likelihood:
$$\ell^{(m+1)} = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{k} w_j^{(m+1)} \phi\left(z_i | \mu_j^{(m+1)}, \Sigma_j^{(m+1)}\right)\right).$$

If $|\ell^{(m+1)} - \ell^{(m)}| > \delta$ for some threshold δ, then return to the expectation step. Otherwise, terminate the algorithm.

In this algorithm, z_s is defined to be the target measurement with index s, prior to assignment to a set Z_j.

After the algorithm has computed the corresponding model, the final association probabilities are used to assign measurements to components, generating the individual sets of measurements Z_j, as defined in Section 3.3.1. This measurement assignment allows further metrics to be computed based on the model and the measurements used to generate it.

One caveat is that one requirement of the EM algorithm is that the number of mixture components is provided to the algorithm. Since part of the task that is being accomplished is identifying the regions of activity, sufficient *a-priori* information to make a good guess as to what the number of components should be will likely not be available. Hence, several models are generated with a different number of mixture components. The *corrected Akaike information criterion* [45] [46] (AICc) is then used to select which of the models to use. The AICc is defined as follows:

$$AICc = 2\kappa - 2\ln L + \frac{2\kappa(\kappa+1)}{n-\kappa-1}, \tag{10}$$

where κ is the number of parameters in the model, L is the maximized value of the likelihood function of the mixture, and n is the number of measurements. The ideal model for a given set of measurements minimizes the loss of information provided by the measurements. That is, the model that minimizes the AICc is the model used to perform resource allocation.

Once the target spread and the target source centers are obtained, the model can be further extended through other observable parameters to aid in resource allocation. For multiple observations of a target, the agent that observes the target will estimate the components of the target velocity. Unlike positional information, this information is *not* shared across agents when sharing measurements: the velocity is used as part of determining the

cost of the agent moving from its current search region to a different search region.

To provide an initial check that the algorithm accurately captures the behavior of a set of data, Figure 6 is the result of running the model computation and selection algorithms on arbitrary data. In this case, the data was generated from three different bivariate Gaussian distributions with 1000 samples per distribution using the following distribution parameters:

$$\mu_{ex1} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T;$$
$$\Sigma_{ex1} = \begin{bmatrix} 10 & 4 \\ 4 & 10 \end{bmatrix};$$
$$\mu_{ex2} = \begin{bmatrix} 14 & 5 \end{bmatrix}^T;$$
$$\Sigma_{ex2} = \begin{bmatrix} 4 & 5 \\ 5 & 20 \end{bmatrix};$$
$$\mu_{ex3} = \begin{bmatrix} -6 & -2 \end{bmatrix}^T;$$
$$\Sigma_{ex3} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}.$$

$(\mu_{ex1}, \Sigma_{ex1})$, $(\mu_{ex2}, \Sigma_{ex2})$, and $(\mu_{ex3}, \Sigma_{ex3})$ are the parameters for each distribution, respectively. Note that both correlated and uncorrelated data were used in the process, indicated by the cross-correlation entries in the covariance matrices. In the figure, the three components are uniquely identified within a degree of error. The resulting calculated component

parameters, including the component weights, are as follows:

$$
\begin{aligned}
w_{ex1} &= 0.341996; \\
\mu_{ex1} &= \begin{bmatrix} -0.02076694 & 0.04622517 \end{bmatrix}^T; \\
\Sigma_{ex1} &= \begin{bmatrix} 10.21450 & 3.644476 \\ 3.644476 & 9.509051 \end{bmatrix}; \\
w_{ex2} &= 0.332052; \\
\mu_{ex2} &= \begin{bmatrix} 13.99328 & 5.055137 \end{bmatrix}^T; \\
\Sigma_{ex2} &= \begin{bmatrix} 3.883991 & 5.044918 \\ 5.044918 & 19.39506 \end{bmatrix}; \\
w_{ex3} &= 0.325952; \\
\mu_{ex3} &= \begin{bmatrix} -6.029838 & -1.970704 \end{bmatrix}^T; \\
\Sigma_{ex3} &= \begin{bmatrix} 1.736427 & -0.09489484 \\ -0.09489484 & 1.836282 \end{bmatrix}.
\end{aligned}
$$

Note that the estimated parameters are accurate within a small degree of error, with a maximum error of 0.7. In addition, the weights indicate that the components have nearly equal probabilities of containing a given sample. This is expected given the independent sampling of the original distributions.

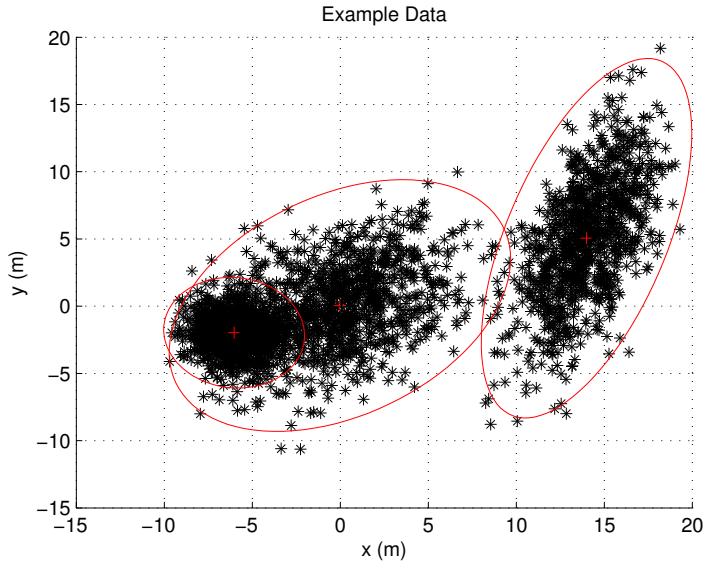


Figure 6. Mixture model generated from arbitrary data. Three independent bivariate Gaussians were used in the data generation process, and they have been uniquely identified.

3.4.2 Model Reduction and Target Streams

While a model properly generated from the measurements and selected will provide a fit for the data, some of the mixture components of the model Q can be very small in volume or have a large degree of separation from the rest of the components in the model. Such components often have very small weights or small covariance volumes, and they can be generated as a result of outlying iceberg measurements that do not associate with a larger component. Running the model through a component reduction algorithm prior to using it can eliminate these components. In addition, as part of detecting when to adjust the regions that are being monitored, mixture reduction can be used to detect the target streams $B_i(t)$, $i \in [1, l]$, that are being generated by ablating sources in U. Keeping track of how many streams are active at a given time can be used to determine when to reallocate agents to new search regions.

3.4.2.1 Model Reduction

Model reduction is a technique that has been primarily studied in the target tracking literature. References regarding mixture component reduction can be found in the target tracking

literature, for the following reason: the general solution of attempting to track a target in clutter in a Bayesian fashion results in a Gaussian mixture. Only a few of the components may actually be the result of target activity; therefore, methods of reducing the number of components in a mixture have been developed to deal with this issue. Most of the extra components have extremely small weights and represent clutter.

Several methods have been devised by researchers to reduce the number of components in a Gaussian mixture; examples include the following:

- Combine components that are similar according to a weighted, squared Mahalanobis distance [47] [48].
- Merge components using gradient descent based on a modified Kolmogorov variational distance, referred to by the investigators as an Integral Square Distance (ISD) [49].
- Combine components that are similar according to a version of Kullback-Leibler discrimination [50].
- Perform mixture reduction through homotopy continuation, formalizing the reduction problem as an optimization problem [51].

As mentioned in [50], aside from using optimization techniques to pick the best mixture distribution, most merging algorithms can be reduced to the following, given a mixture with n components to be approximated by a mixture of m components, and while more than m components remain:

1. Determine measures of similarity between all components in the mixture.
2. Depending on the measure of similarity, for each pair of similar components, replace the pair with their moment-preserving, merged component.

The end result is a mixture distribution whose overall mean and covariance (*i.e.*, a weighted combination of the means and covariances of the individual components) is effectively the same before and after the mixture reduction process.

A moment-preserved component is formed by combining the weights, means, and covariances of a component pair in the following manner, using the notation of [50] and given that the indices of the components of interest are i and j:

$$w_{ij} = w_i + w_j, \tag{11}$$

$$w_{i|ij} = \frac{w_i}{w_i + w_j}, \tag{12}$$

$$w_{j|ij} = \frac{w_j}{w_i + w_j}, \tag{13}$$

$$\mu_{ij} = w_{i|ij}\mu_i + w_{j|ij}\mu_j, \tag{14}$$

$$\Sigma_{ij} = w_{i|ij}\Sigma_i + w_{j|ij}\Sigma_j + w_{i|ij}w_{j|ij}(\mu_i - \mu_j)(\mu_i - \mu_j)^T, \tag{15}$$

where w_{ij} is the combined weight of components i and j, $w_{i|ij}$ and $w_{j|ij}$ are the scaled weights of components i and j, μ_{ij} is the combined mean of the components, and Σ_{ij} is the combined covariance matrix.

Note that when merging components, the new mean is the weighted sum of the means of the component pair, while the new covariance matrix requires a spreading of the means term in addition to the weighted covariance matrix sum.

For the purposes of *in-situ* modeling, an algorithm that converges relatively quickly and does not result in anomalous behavior in certain corner cases is desired. While optimization techniques may not have issues in convergence speed, the result may vary depending on how the algorithm was initialized, which can be a tricky process. Hence, the methods of Salmond [48] and Runnalls [50] will be candidates for algorithmic components to be incorporated into a model-processing pipeline. The intent of the result of the pipeline is to be able to determine whether or not streams of targets have vanished, hence, complete accuracy is not necessary; the mixtures only need enough reduction to remove spurious components.

A comparison between the two reduction techniques may be conducted by analyzing simulated data that consists of measurements of target positions from multiple streams of targets $B_i(t)$ from ablating sources. Figures 7 and 8 are two plots of example mixtures generated from a set of such data. Target measurements are marked with asterisks, and the covariance ellipses of each of the individual components are the red ellipses. The component means are the red crosses in the centers of the covariance ellipses. Note that the results of generating the mixtures clearly distinguish the four streams, with one stream that is associated with two components. The three-stream example has one stream that is covered by three components. The application of a mixture reduction technique can generate a mixture that merges some of these "extra" components.

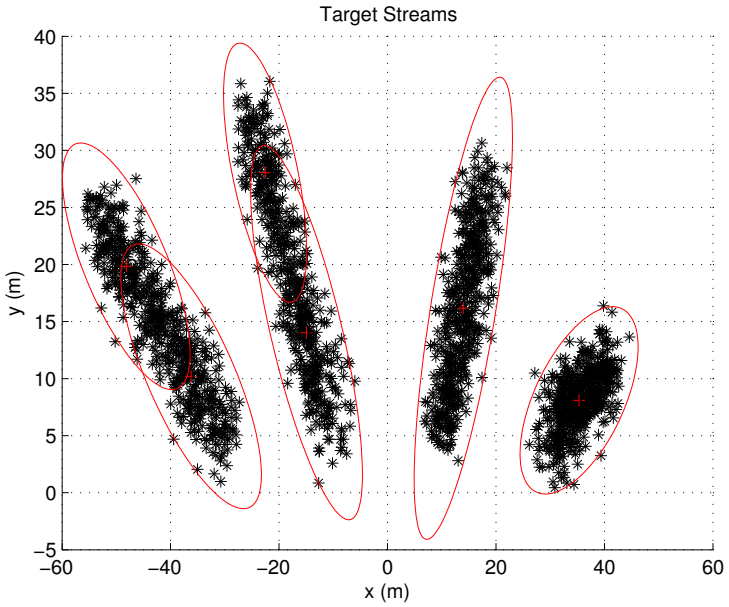


Figure 7. Mixture model for four target streams. Note that all target streams have been identified.

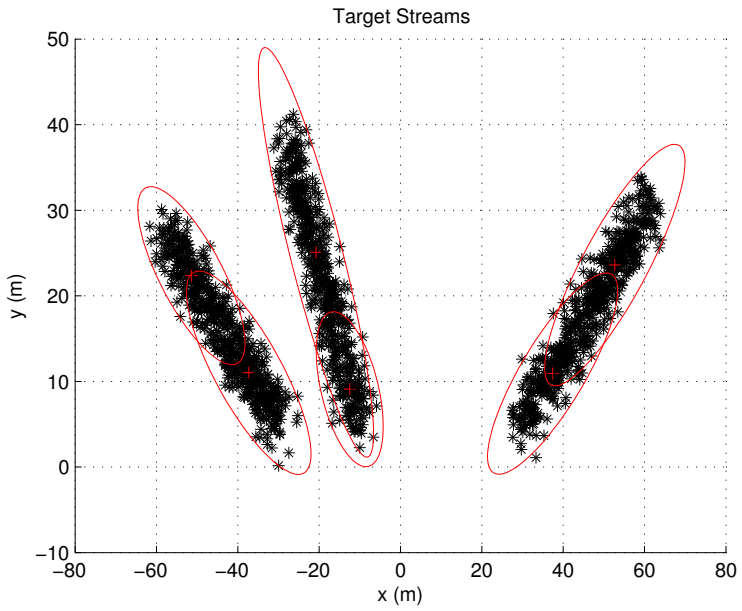


Figure 8. Mixture model for three target streams. Note that all target streams have been identified.

For the two data sets, they were reduced using both the Salmond and Runnalls algorithms. The results for each of the two data sets for both mixture reduction algorithms were, in fact, identical; the results from using the Salmond algorithm are shown in Figures 9 and 10.

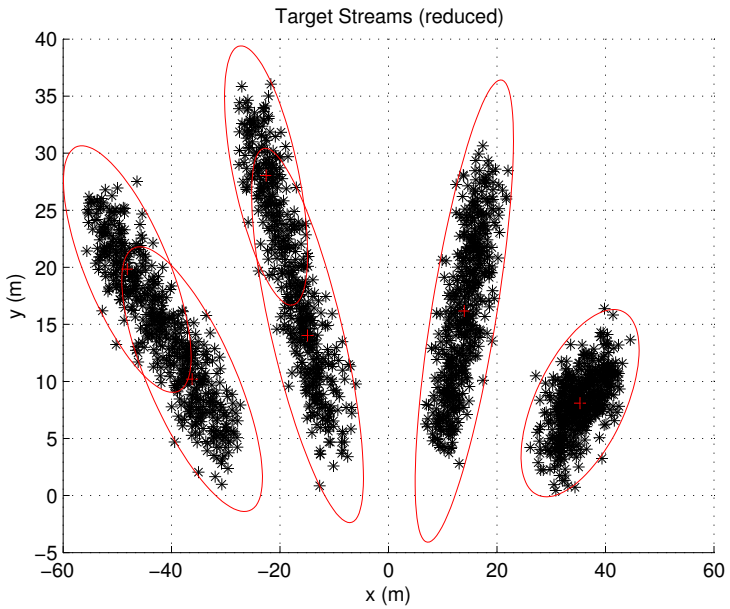


Figure 9. Reduced mixture model for four target streams (Salmond algorithm).

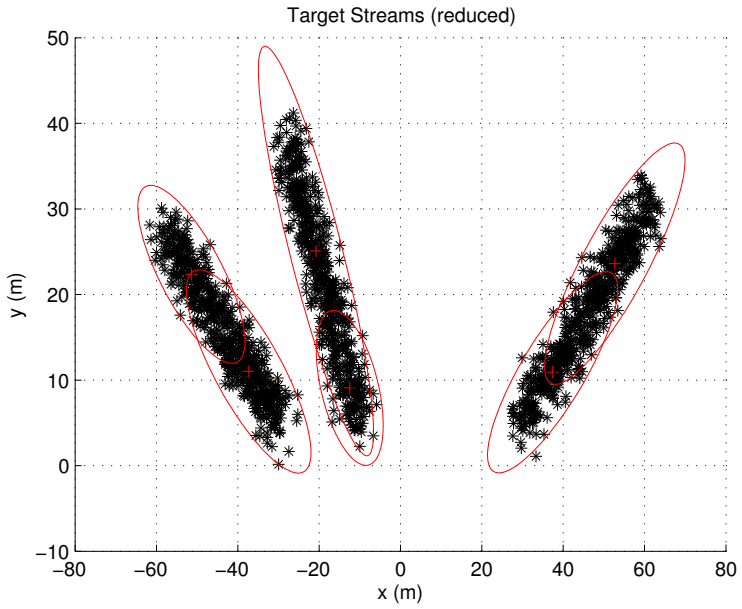


Figure 10. Reduced mixture model for three target streams (Salmond algorithm).

In both of these cases, it can be observed that the individual target streams were again identified. However, for this particular data set, which is the type that one would expect for observations of iceberg position, there is very little difference between the performance of

the two algorithms. That is, the observations are tightly clustered and the resulting components have relatively high magnitude weights. It may be concluded, from these results, that which type of reduction algorithm should be used depends on the type of data that is encountered. If preservation of the overall mean and covariance of the resulting model is necessary, then the Runnalls algorithm should be used. However, in practice for the types of models that are generated and the eventual use of these models (generating search regions for mobile sensors), preserving the overall mixture parameters is not paramount. Since the Salmond algorithm relies more on the separation of means, and spatial properties are more of interest here, for most mixture reductions, that algorithm is sufficient.

3.4.2.2 Component Clustering

Once a reduced mixture has been obtained, it may still not be suitable for detecting the target streams $B_i(t)$. For example, a component with a weight that prevented it from being merged may reside within the covariance volume of another component. This would be a spurious result if one were to use the components to determine the number of target streams.

One solution to the problem is to execute a clustering algorithm on the remaining components, ignoring whether or not the overall mean and covariance is changed, since the primary concern at this point is the components which provide a target stream representation, rather than search region representations. However, to compare components, a distance measure of some kind is necessary. In this case, a quantity that measures the separation or overlap between two distributions is desired.

One such quantity is the Bhattacharyya coefficient [52]. The Bhattacharyya coefficient has seen use in computer vision (*e.g.*, [53]) and in signal processing applications.

The coefficient is derived as part of the Bhattacharyya distance, which measures the similarity between two distributions; the distance $D_B(p, q)$ between any two probability distribution functions $p(x)$ and $q(x)$ is defined as follows:

$$D_B(p, q) = -\ln\left(BC(p, q)\right), \tag{16}$$

where $BC(p, q)$ is the Bhattacharyya coefficient. The coefficient takes varying forms depending on whether the distribution is discrete or continuous. As the distributions of interest are Gaussian, the continuous form is necessary:

$$BC(p, q) = \int \sqrt{p(x)q(x)}\, \mathrm{d}x. \tag{17}$$

The coefficient ranges from zero to one, depending on the amount of overlap. For two fully overlapping distributions, the coefficient will be one. The continuous form of the coefficient can be difficult to evaluate, but if both $p(x)$ and $q(x)$ are both multivariate Gaussian probability distribution functions, as is the case here, a closed form exists [52] for the Bhattacharyya distance:

$$D_B(p, q) = \frac{1}{8}(\mu_p - \mu_q)^T \Sigma^{-1} (\mu_p - \mu_q) + \frac{1}{2} \ln \left(\frac{\det \Sigma}{\sqrt{\det \Sigma_p \det \Sigma_q}} \right), \tag{18}$$

where

$$\Sigma = \frac{1}{2} \left(\Sigma_p + \Sigma_q \right), \tag{19}$$

and (μ_p, Σ_p) and (μ_q, Σ_q) are the means and covariances of $p(x)$ and $q(x)$. The Bhattacharyya coefficient then follows trivially from the definition of the distance, as follows:

$$BC(p, q) = \exp\left(-D_B(p, q)\right). \tag{20}$$

As part of a clustering algorithm, a straightforward design for the algorithm that incorporates the coefficient will be used. First, a threshold ζ on the Bhattacharyya coefficient is set. As the coefficient decays exponentially the further the distributions under consideration are apart, a small ζ is sufficient to indicate that the components are sufficiently close; for this research, a $\zeta = 0.01$ is used. Next, a component is selected from the list of components to act as the initial cluster head. Then, the following steps are performed until all of the components have been assigned to a cluster:

1. The Bhattacharyya coefficient is computed between the current cluster head and a component.

2. If the component distance is *greater* than the threshold, since the coefficient approaches one as the distributions overlap, that component is assigned to the cluster.

3. If all clusters are exhausted and the component has not been assigned, a new cluster with this component is created as the head.

4. The next component is selected from the list and the algorithm is re-iterated.

The resulting set of component clusters, given sufficient separation between the original components, should then represent the streams of targets that are present. This algorithm can then be incorporated into a processing pipeline for target measurements.

3.4.2.3 Model Pipeline

Given the previously described algorithms for generating a model and extracting additional data from it based on the model components of Q, a complete algorithmic pipeline can be developed for iceberg measurements. Note that this complete pipeline is only intended for the local, per-sensor iceberg model, so that each mobile sensor has the capability to assist in reallocation of sensor resources. Additionally, attempting to execute this pipeline on the global model will not produce the desired results, unless the components are well-separated, which may not be the case.

The complete pipeline for processing measurements of iceberg target positions is then as follows:

1. Using local iceberg position measurements, the sensor agent computes and selects an appropriate Gaussian mixture model using a combination of expectation-maximization coupled with the corrected Akaike information criterion.

2. The sensor agent computes and stores metrics from the model.

3. The sensor agent then runs a mixture reduction algorithm over the model such that the overall mean and covariance of the entire mixture are unchanged.

4. Finally, the sensor agent clusters the remaining components using the Bhattacharyya-coefficient-based clustering algorithm.

The result will be a handful of mixture components that represent individual target streams, which can then be used to keep track of their number. Once the number of streams or other properties of the streams change, then appropriate actions may be chosen, such as transmitting an indicator that sensor resources should be reallocated.

3.4.3 Metrics

To perform resource allocation, the model must first be scored and metrics computed. With these metrics, the most effective agent configuration can be determined. In addition, an appropriate set of metrics can be used to predict information about a given model; specifically, based on past observations, the behavior of the model, and as a consequence iceberg drift, can be predicted in terms of its metrics. This predicted information can be used in producing a forecast, or in adjusting sensor parameters to perform more effective observation of an area.

Metrics are calculated by each of the agents based on their own individual pictures of the model, and used as part of determining costs for resource allocation. The set of metrics that will be defined for these models are as follows:

Number of mixture components: The number of components contained within the GMM generated from a set of iceberg position measurements obtained by a group of agents. This number represents the potential number of search/ice-ablation regions that may be present. The number of components depends entirely on the target measurements: their number and spatial diversity. With greater spatial diversity, the number of components can increase.

Mean acquisition time: The time to acquire the initial contact on a target averaged across all target measurements. This metric is the key quantity for minimizing the

objective function. This metric will increase with greater total model area depending on target velocity and agent velocity.

Total model area: The area covered by the model when all of the components are taken into account. This area can vary depending on how many sigmas are used in computing the associated error ellipses of each GMM component; in general, the area will vary in these terms from one sigma to three sigma. This metric can be used in determining additional coverage metrics, and as part of determining weighting for the cost function. The area increases with the volume of the mixture components, which varies directly with the covariance matrices of the components.

Number of acquired targets: The number of unique targets that have been acquired during an observation task. This metric is used in characterizing the overall performance of a given search algorithm. This metric will vary with target velocity, agent velocity, sensor field-of-view (FOV), and source-ablation rate. Generally, it will increase with larger fields of view and higher source-ablation rates. However, with large agent velocities comparable to target velocity, the number of acquired targets could decrease, with small sensor FOV.

Number of targets acquired per agent: The number of unique targets that have been acquired per agent; this is not an average, this metric is in fact a set of numbers. This metric is used to determine agent loading; *i.e.*, whether or not a region should be split amongst multiple agents if an agent is determined to be overloaded. The same values that affect the total number of acquired targets affect the number of targets acquired per agent.

Average target velocity: The average target velocity over all measurements obtained by all agents. This metric is used to determine search speeds and sensor fields of view. The target velocity will correlate with the number of acquired targets.

Agent-specific target coverage: The percentage of targets obtained over all agents over a given amount of time. This metric is used to determine performance of a particular search algorithm, if it can be computed. To compute this metric, truth data is required. This metric will generally increase with greater sensor FOV, greater numbers of agents, and lower target velocities. Total model area will correlate with the target coverage.

Predicted number of agents needed to cover an area: This derived quantity is the number of agents required to effectively search a given area and provide sufficient region coverage. This metric is determined by a combination of previous metrics, and it is needed to determine how many agents are available to allocate to particular search regions. This metric will increase with total model area, and it will decrease with sensor FOV.

Total number of agents required for a given model: This derived quantity is the total number of agents required to cover the total area of a particular GMM. This can vary based on several factors, especially the total model area metric. This metric is used to characterize algorithm performance and the "difficulty" of a particular model in terms of required resources. This metric will increase in magnitude with total model area and number of mixture components, and decrease with sensor FOV.

With the metrics defined, they may then be expressed in terms of expectations with respect to model parameters. The symbols and equations for each metric are defined in Table 1.

The metrics can be examined in terms of how they affect one another; performing such an analysis is key in determining how a model can be predicted to evolve over time.

Table 1. Metrics descriptions and definitions.

Description	Definition
Number of mixture components	k
Mean acquisition time	$T_{s,mean} = E[T_s \vert Z_j]$
Total model area	$A_{m,t} = \sum_k A_k$
Number of acquired targets	$N_T(T)$
Number of targets acquired per agent	$N_A(T)$
Average target velocity	$\bar{v}_T = E[\bar{v}_j \vert Z_j]$.
Agent-specific target coverage	$C_A(T) = \frac{\sum N_A(T)}{N_T(T)}$
Predicted number of agents needed to cover an area	M_A
Total number of agents required for a given model	M_m

3.5 Example Models Using IIP Data

To validate the modeling approach and illustrate the associated metrics that result from using the previously described modeling techniques, a selection of data from the IIP yearly sightings database reports, which are available online [54], has been analyzed using this methodology [55].

3.5.1 Model Generation

Mixture models were generated from the database reports provided by the IIP. Since each yearly data set is quite large, to provide a fair comparison of the models between years, iceberg sightings from the region shown in Table 2 were extracted from each of the data reports prior to running the modeling algorithms. A map of this region is shown in Figure 11. As suggested by the extents of the region of interest, the coordinates of the iceberg sightings are recorded as latitude and longitude. To ensure that the scaling is correct in the resulting models, the coordinates are first converted to Universal Transverse Mercator

Table 2. Parameters of the region used in the IIP data set studies.

Coordinate	Minimum	Maximum
Latitude	55.0°	57.0°
Longitude	−59.8°	−58.4°

(UTM) coordinates, using the equations as described in [56]. Prior to computing the models, the coordinates are then adjusted to be relative to the center of the region of interest.

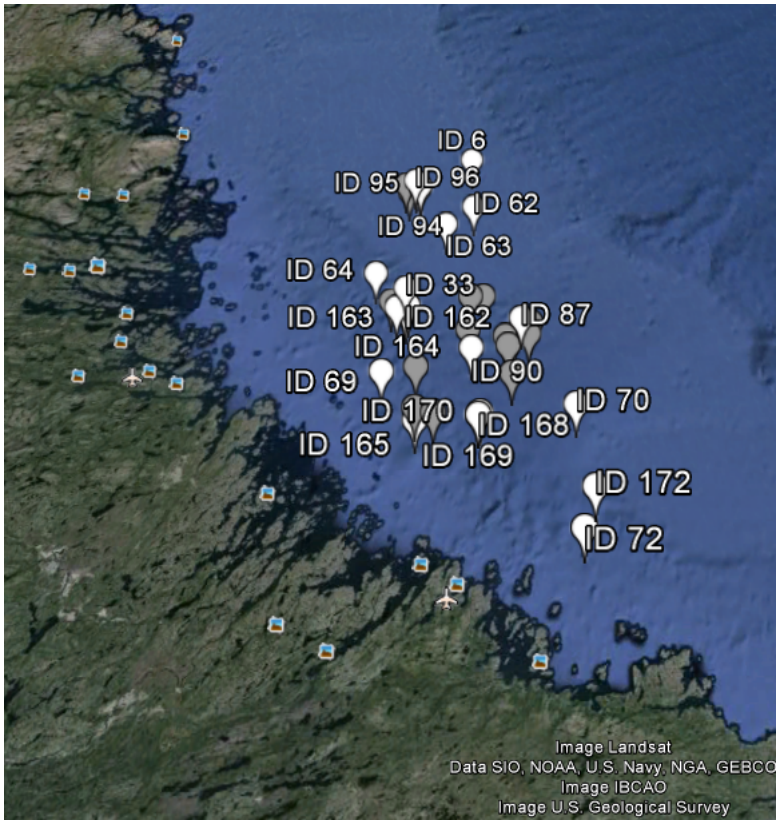


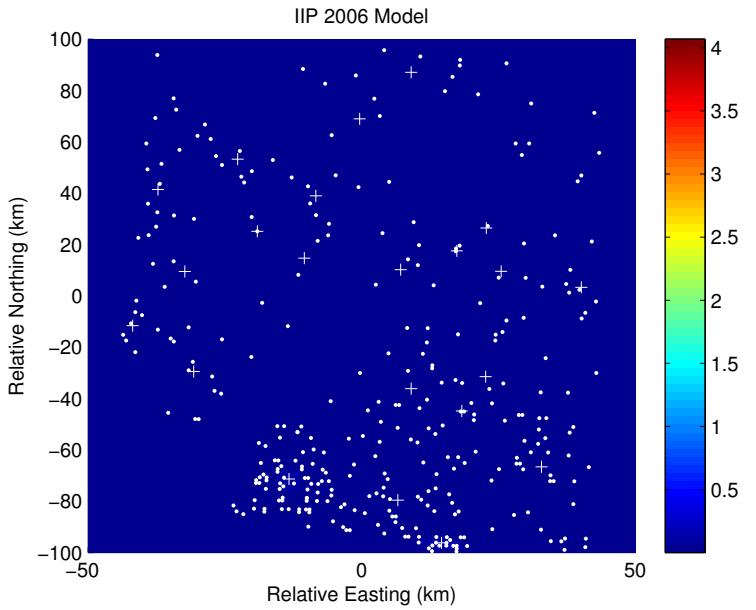
Figure 11. Map of region used in the IIP data set studies (Google Earth). The points used are from the 2007 data set.

Figures 12, 13, 14, 15, 16, and 17 illustrate mixture models generated from each of the IIP data sets from the years 2006 to 2011, inclusive.[2] The x and y axes on the plots are iceberg sighting relative UTM easting and northing, respectively. The white dots are the

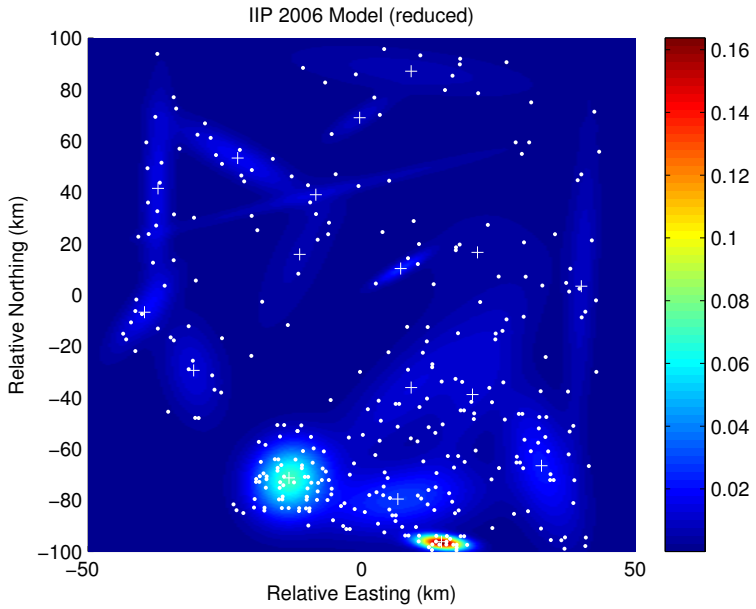
[2]Data from previous years have not been used as part of this study; the data sets are either incomplete, or do not contain sightings within the region of interest.

iceberg sightings, while the white crosses are the centers of the resulting mixture components. Up to 25 components were allowed as part of each mixture distribution. As a result of the nature of the data; *i.e.*, its high spread, the Salmond mixture reduction algorithm was used to see an appreciable difference between the reduced and non-reduced mixture distributions.

Note that once computed, for these particular results, the magnitude of the metrics will be very large. In an actual mission when these models are being generated in real-time, the areas will be much smaller, since the overall region of interest will be closer to a specific glacier, rather than examining the behavior of icebergs that have already been calved and are adrift at sea. One can think of analyzing the IIP data as examining a long-duration, global case, while real missions are focused exclusively on a glacier-specific local case.

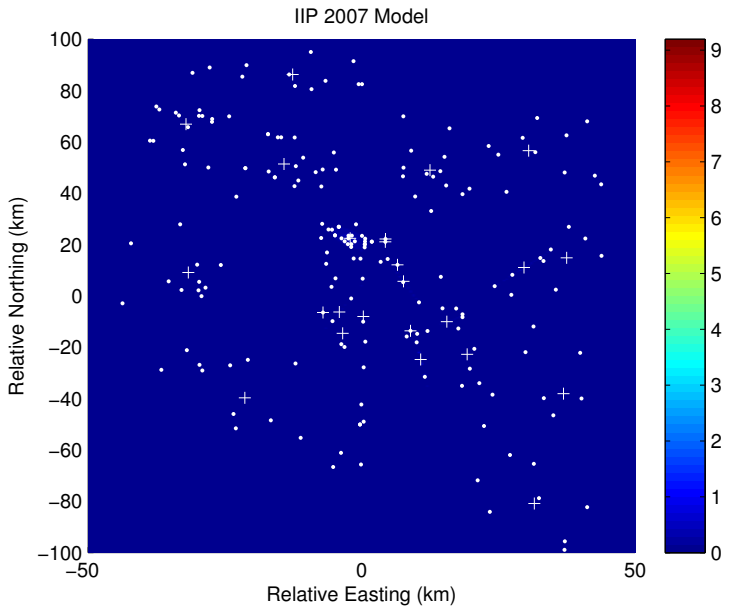


(a) Full Model

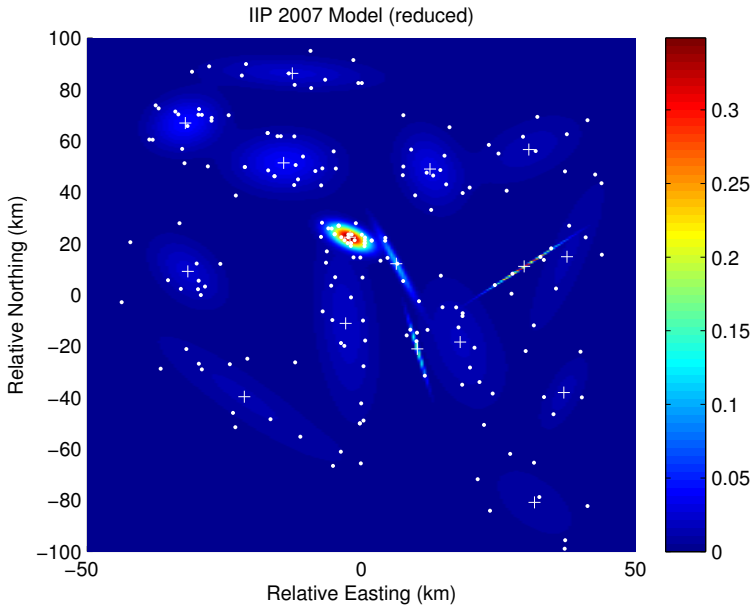


(b) Reduced

Figure 12. Mixture model generated from the 2006 IIP data set.

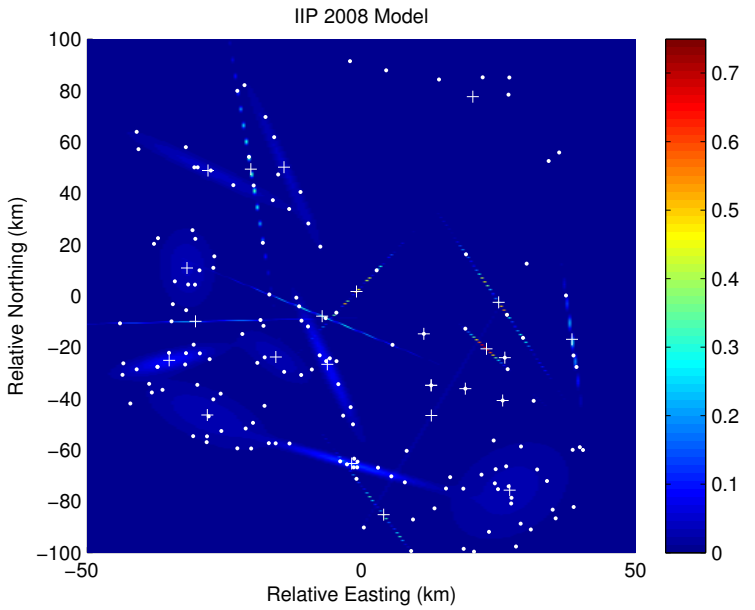


(a) Full Model

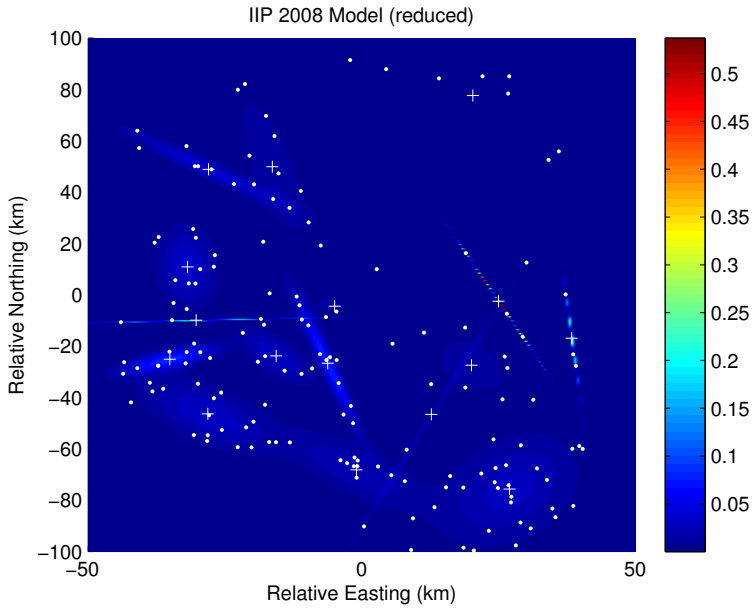


(b) Reduced

Figure 13. Mixture model generated from the 2007 IIP data set.

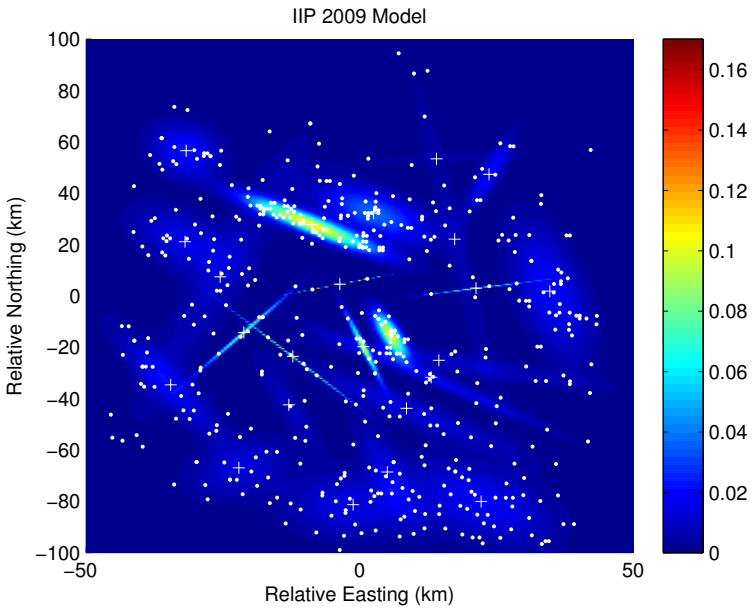


(a) Full Model

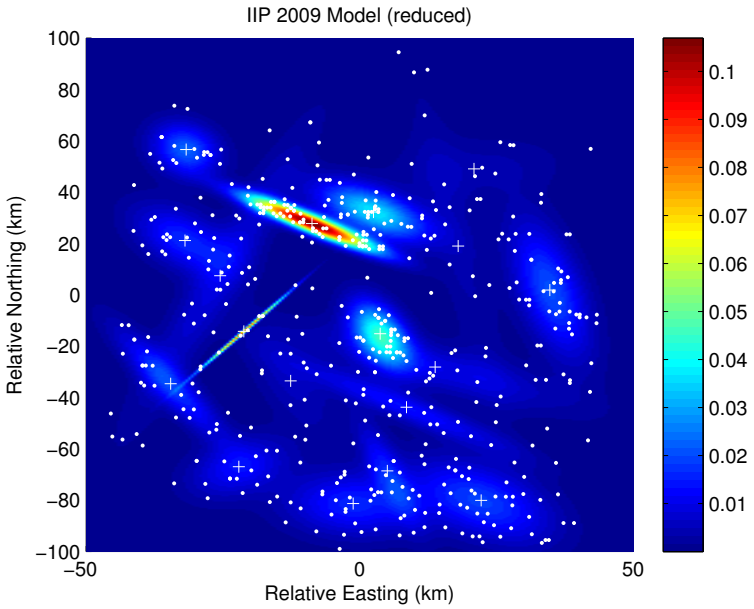


(b) Reduced

Figure 14. Mixture model generated from the 2008 IIP data set.

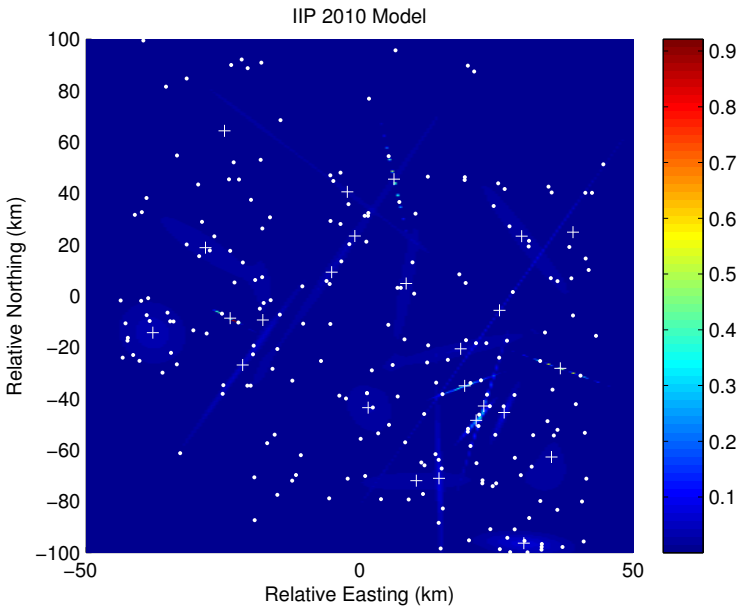


(a) Full Model

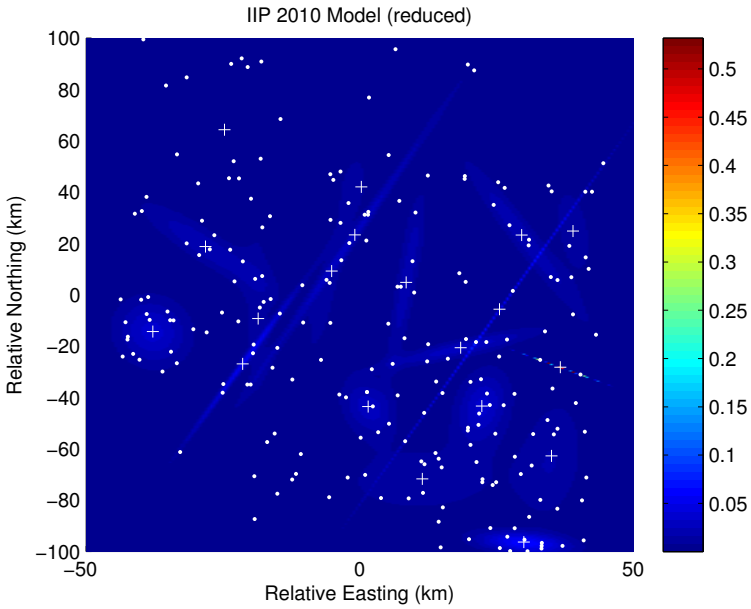


(b) Reduced

Figure 15. Mixture model generated from the 2009 IIP data set.

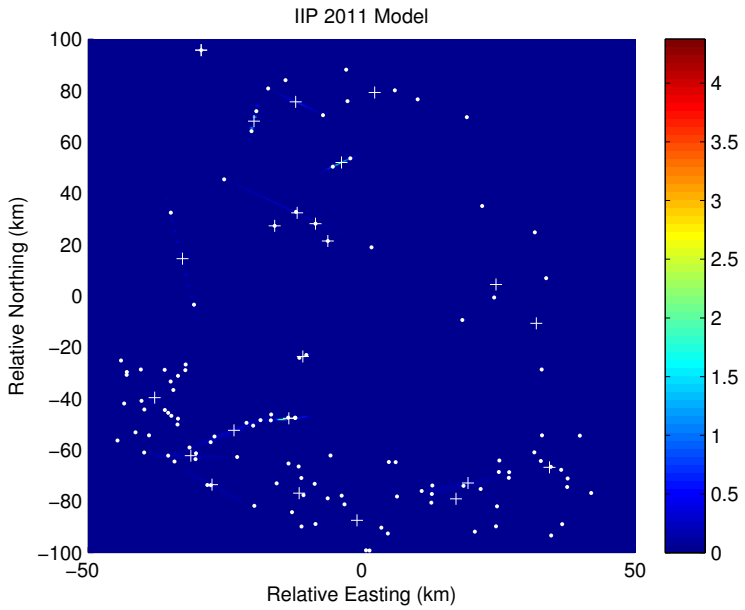


(a) Full Model

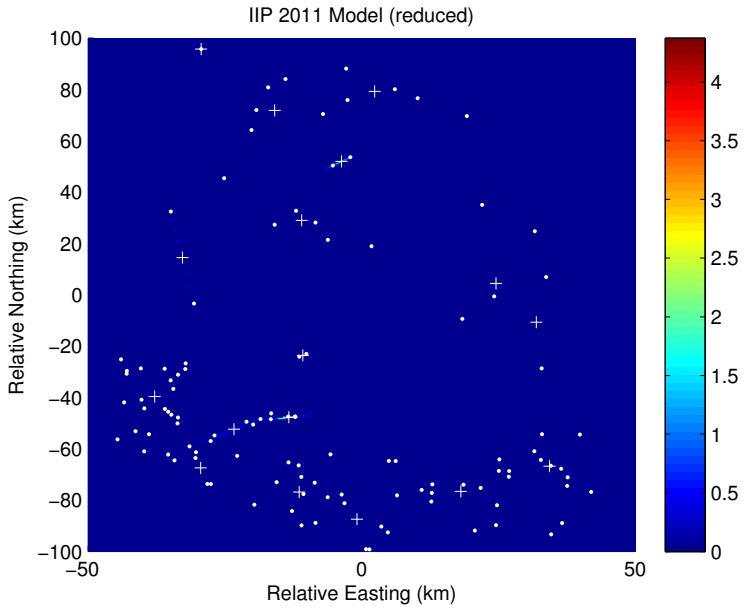


(b) Reduced

Figure 16. Mixture model generated from the 2010 IIP data set.



(a) Full Model



(b) Reduced

Figure 17. Mixture model generated from the 2011 IIP data set.

In each of the figures, regions of low target-likelihood and high target-likelihood are present, indicated by the intensity of each of the heat maps. These regions correlate directly

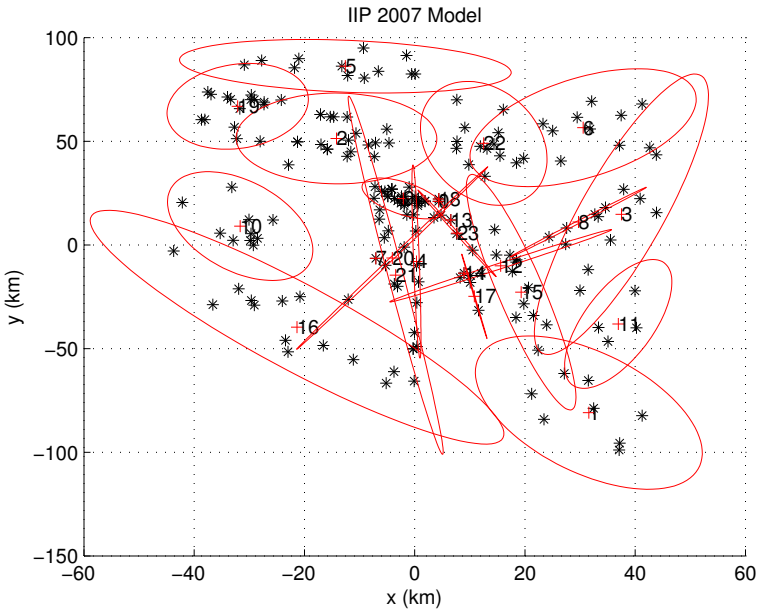
with target density. It is also clear that for these particular data sets, mixture reduction provides a benefit for interpreting various regions of activity. In several cases, particularly in the years 2007 (Figure 13) and 2008 (Figure 14), the structure in the model is made more apparent by applying mixture reduction to the results. Additionally, if the results of these modeling experiments were to be used as part of initializing the search algorithms for a set of mobile sensors, reducing the mixtures allows for the generation of a smaller number of regions that should be searched, which can provide for a reduction in the resources necessary to observe the region of interest.

Visual representation of the target densities provides some insight on the behavior and spatial density of the icebergs for a given year. In 2009, shown in Figure 15, many of the icebergs were sighted in very dense regions, while in 2006 and 2007, the spread was much greater and only a few regions of high target density exist. However, 2006 is far less sparse in overall model structure than 2007. The regions of high target density in this model may be indicative of either higher ablation rates when closer to land or clusters of icebergs being trapped in ocean currents when further at sea.

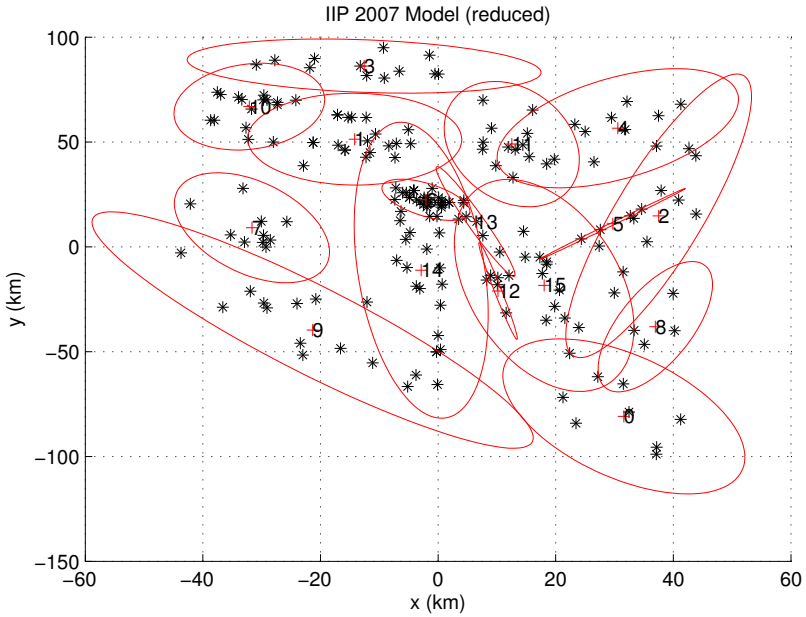
It should be noted that a qualitative evaluation of the models in this manner does not necessarily provide an accurate picture of the behavior of the icebergs during each of these years, especially if the observations are very sparse, as in the case of 2011 (Figure 17). This sparsity will result in the generation of a large number of highly separated mixture components with very small weights, and thus structure can be lost if only examining the resulting map of likelihoods. Another example of this phenomenon is in the year 2007, where there are two strong components of high target density, while the others have much smaller weights. If the 3-σ error ellipses for 2007 are plotted, as in Figure 18, discrete regions of activity can in fact be observed, as in the reduced model shown in Figure 13(b), but when comparing the ellipses with the likelihood map, almost all of the iceberg probability is clustered into two components that are smaller in volume with respect to the rest of the observations. This is suggestive of the existence of a subregion in the region of interest

where ocean currents trapped icebergs. Comparing the mixture model of 2007 shown in Figure 18(a) with its reduced counterpart Figure 18(b), only the components close to the component containing most of the probability were merged, but the combined weights did not substantially influence the overall distribution of probability, emphasizing the influence of this smaller subregion.

Given these qualitative analyses, metrics computed using the models and compared with the pictures of the likelihood functions can provide a more uniform picture of iceberg behavior for each of these years.



(a) Full Model



(b) Reduced

Figure 18. 3-σ error ellipses for the 2007 IIP data set.

3.5.2 Metrics

With the models computed in the previous subsection, metrics are computed from the results. Note that not all of the metrics as defined in Section 3.4.3 can be computed here, as some of them require ground truth, which is unavailable from the sighting database; *e.g.*, coverage. Therefore, the following metrics are computed for each of the models:

- Number of acquired targets.
- Average target velocity.
- Total model area.
- Number of components.

It should also be noted that the average velocity is the best estimate given the data, as the identifiers that are assigned to an iceberg are not necessarily unique. In many cases, however, it is reasonable to assume that an identifier that is the same between two sightings in the database that are reasonably close in time refers to the same iceberg.

The metrics computed from the models are summarized in Table 3. A few observations can be made from this data:

1. The number of sightings increases considerably in the years 2006 and 2009.
2. The average target velocity has variability between years, yet stays within a range.
3. The spread of the icebergs, as illustrated by the total model area, is similar for all years except 2007 and 2011.

However, these observations have some caveats. The number of sightings could increase as a result of better algorithms for analyzing and classifying targets identified from a satellite SAR image, as an example. In addition, models where the maximum number of components provides the best fit to the data often have single components devoted only to a small handful of target sightings.

Table 3. Metrics for the analyzed IIP data.

Year	N_T	$\bar{v}_T$	$A_{m,t}$	K
2006	331	0.106 m/s	101,133 km^2	22
2007	193	0.181 m/s	57,063 km^2	24
2008	155	0.199 m/s	86,351 km^2	25
2009	517	0.184 m/s	114,044 km^2	24
2010	214	0.175 m/s	176,970 km^2	25
2011	115	0.147 m/s	51,670 km^2	24

What is more useful from this set of metrics is the velocity and area estimates: thus far, from six years of observations, a maximum (2010) and minimum (2011) iceberg spread in terms of area can be characterized. Additionally, some periodicity can be observed in the magnitude of the area as the icebergs advance and recede over the years of interest. The velocity is relatively stable for this region, except for two years. Such behavior is not atypical; the velocities of surface ocean currents vary considerably from year to year. Incorporating other years of IIP iceberg sightings, and fusing the models would provide a better picture of past, present, and future iceberg behavior.

3.5.3 Summary

From the results presented in this section, the proposed method of statistical modeling can be beneficial in understanding certain aspects of iceberg behavior. This modeling method can be used in determining forecasts of future iceberg behavior based on metrics computed from generated models or to predict regions of new iceberg activity. It is this latter prediction that is the ultimate goal of this methodology, as it allows for defining regions that have a higher focus placed upon them for acquiring new targets, and thus minimizing the objective function: target acquisition time. Therefore, after model generation, new search regions must be extracted from the model.

3.6 Search Region Extraction

Having generated a model based on the iceberg sightings, a method of extracting the search regions that a set of agents should use to sufficiently explore the activity regions must be

devised. Such a method can be developed from the standard method of generating an error ellipse or contour of constant probability from the mean and covariance of a distribution. In this case, the error ellipses will be generated from each of the components of the mixture distribution, which can then be assigned to particular agents for search. Each of the agents can then use a search pattern [39] [40] to scan the search regions.

To extract the search regions and determine the new search pattern parameters, each mixture component is considered to be a separate probability distribution in its own right. The steps in an algorithm to extract the search regions may then be defined as follows:

1. Calculate parameters for the contour of constant probability from the covariance matrix: the parameters of the contour computed using the parameters of the distribution can be used to determine an appropriate search area.

2. Fit a rectangle to the probability contour: for flexibility in choosing search patterns, a rectangular search region is fitted to the contour of constant probability, as most well-known search patterns adapt well to this type of search area.

3. Calculate new search-pattern parameters: these new parameters will allow the search pattern to appropriately cover the new search area.

The following sections describe these steps in greater detail and provide a method by which the contour of constant probability may be adaptively resized. In addition, limitations that must be taken into consideration when extracting these regions are described.

3.6.1 Contour of Constant Probability

The contour is characterized by the eigenvalues λ_i and eigenvectors $\mathbf{e}_i$ of the covariance matrix. Both sets of these parameters fully describe the axes of the resulting ellipsoid; *i.e.*, the contour is defined where vectors $\mathbf{x}$ satisfy the relation

$$(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu) = c^2, \tag{21}$$

which is related to the Mahalanobis distance. Specifically, $\mathbf{\Sigma}$ is the covariance matrix, μ is the mean, $\mathbf{x}$ is the vector to compare, and c is a constant Mahalanobis distance.

The axes of the ellipsoid have the form $\pm c\sqrt{\lambda_i}\mathbf{e}_i$. The constant c is chosen using a chi-square distribution χ_k^2 such that $c^2 = \chi_k^2(\alpha)$, as this leads to a contour containing $(1-\alpha)\times 100\%$ of the probability. The number of degrees of freedom k of the chi-square distribution is equal to the number of dimensions of the multivariate-Gaussian distribution.

3.6.2 Fitting a Rectangle

Once the axes of the probability contour have been determined, a search region can be generated from the axes. A rectangle is fitted to the contour as stated in Section 3.6 and the interior of the rectangle is used as the search region. Ample region coverage with the search patterns to be discussed will be obtained as a result.

To fit the rectangle, the two extreme points of the ellipse are obtained and are used as the basis points. First, the maxima and minima of the parametric equations of the ellipse must be computed. The ellipse equations are the following:

$$x(t) = x_\mu + a\cos t\cos\phi - b\sin t\sin\phi; \tag{22}$$

$$y(t) = y_\mu + a\cos t\sin\phi + b\sin t\cos\phi, \tag{23}$$

parameterized by the time t, where x_μ and y_μ are the components of the point at the mean, and ϕ is the rotation angle with respect to the x-axis of the major semiaxis a. This angle may be found via the following equation:

$$\phi = \frac{1}{2}\tan^{-1}\left(\frac{2\sigma_{xy}}{\sigma_x^2 - \sigma_y^2}\right), \tag{24}$$

where σ_{xy} is the off-diagonal cross-covariance from the covariance matrix, and the two σ terms are the variances of the x and y coordinates. To obtain the appropriate extrema, the ellipse equations are differentiated

$$\frac{dx(t)}{dt} = -a\sin t\cos\phi - b\cos t\sin\phi; \tag{25}$$

$$\frac{dy(t)}{dt} = b\cos t\cos\phi - a\sin t\sin\phi, \tag{26}$$

and set equal to zero, which results in the following expressions for t:

$$t_x = \tan^{-1}\left(-\frac{b}{a}\tan\phi\right); \tag{27}$$

$$t_y = \tan^{-1}\left(\frac{b}{a}\cot\phi\right). \tag{28}$$

The t value that gives the maximum is found with the appropriate substitutions for a, b, and ϕ; $t + \pi$ provides the minimum. The points are obtained by substituting the values of t into the original parametric equations $x(t)$ and $y(t)$. The resulting set of rectangle corner points is $\{(x_{min}, y_{min}), (x_{min}, y_{max}), (x_{max}, y_{max}), (x_{max}, y_{min})\}$. An illustration of the result is shown in Figure 19. Despite the fact that the figure shows an angle of rotation of the ellipse indicating correlation between the coordinates, the resulting rectangle has the same corner points regardless of any cross-covariance terms in the covariance matrix.

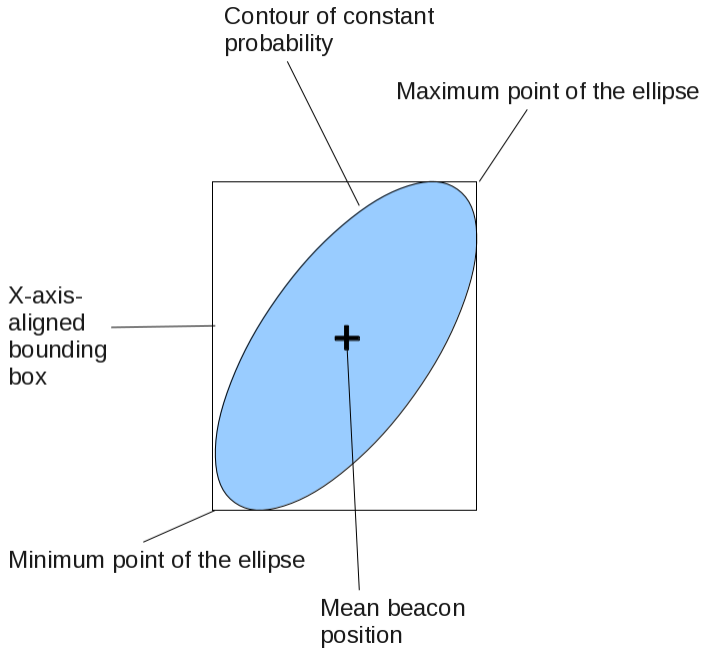


Figure 19. Fitting a rectangle to a contour of constant probability.

3.6.3 Search Pattern Parameters

With the points that characterize the polygon that acts as the search region, the search pattern parameters may be modified to constrain the extents of the search to this new region.

Two search patterns will be considered in this section: the parallel-transect or "lawnmower" search pattern and the arithmetic spiral pattern. Illustrations of these search patterns are shown in Figure 20. The pattern at the top illustrates the arithmetic spiral, while the pattern at the bottom illustrates the parallel-transect pattern.

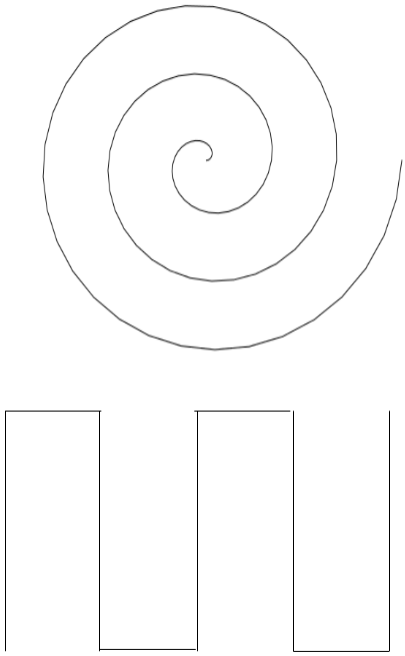


Figure 20. Examples of search patterns. Top: arithmetic spiral pattern. Bottom: parallel-transect/lawnmower pattern.

In the case of the parallel-transect search pattern, the horizontal distance is set to the range of the sensor used to scan for targets, which will be defined to be d_s.

The vertical distance traversed by the agent is set to $D_v = |y_{max} - y_{min}|$, and the number of search-pattern traversals N_{trav} is set to

$$N_{trav} = \left\lceil \frac{|x_{max} - x_{min}|}{d_s} \right\rceil, \tag{29}$$

where a traversal is defined to be one vertical path followed by a horizontal path.

The spiral pattern has fewer degrees of freedom: only the furthest extent of the spiral can be fitted to the spatial representation of the distribution. The standard equation for the

arithmetic spiral in polar coordinates is the following:

$$r(\theta) = a + b\theta, \tag{30}$$

where a sets the orientation of the spiral, and b sets the spacing between arms of the spiral. As it is a cosmetic parameter, a is set to zero. To properly set b, first note that the distance between arms of the spiral is $2\pi b$. $2\pi b$ must be at least d_s to have a similar detection capability as that of the parallel-transect pattern while traveling along the curve of the spiral. Hence, $b = d_s/2\pi$.

From the polar definition of the spiral, a parametric representation is derived:

$$x(t) = \alpha t \cos \xi t; \tag{31}$$

$$y(t) = \alpha t \sin \xi t, \tag{32}$$

where $\alpha = b\xi$. ξ is a design parameter that sets the spacing in angular units between individual points on the spiral; ξ may be considered a number effectively equivalent to a frequency. For the experiments conducted throughout this research, the value of ξ is set to 25°.

Next, the spiral-pattern parameters are computed for the reduced search area defined by the rectangle. For the spiral, only the final value of t, t_f, can be manipulated:

$$t_f = \frac{\sqrt{|y_{max} - y_{min}| \cdot |x_{max} - x_{min}|/\pi}}{b\xi}. \tag{33}$$

The final value becomes the initial value of t when traveling in a reverse direction along the spiral. In a practical implementation, the maximum of the two distances between the coordinates is chosen and is used in calculating the extent; *i.e.*,

$$t_f = \frac{\sqrt{d_{max}^2/\pi}}{b\xi}, \tag{34}$$

where d_{max} is that maximum.

3.6.4 Contour Sizing

A design choice may be made with respect to the error ellipses computed from the mixture model: to fully cover the active regions of a particular glacier, should additional area be added to the search region? Or, is the extent such that the area that should be searched is much smaller than expected; *e.g.*, ocean currents are such that the iceberg velocities have vectors that take them through a certain region at a slow rate, requiring less effort to detect them. One solution to this problem is to adjust the number of sigmas used to generate the error ellipse based on the measurements; *i.e.*, change the percentage of probability contained within the probability contour based on the statistical distance of the measurements from the mean, as opposed to the fixed-size search regions that are generated by the algorithm described in Section 3.6. The Mahalanobis distance of samples from the estimated distribution can be used to determine the appropriate size of the search region. The following formulation of this distance is used:

$$d = (\mathbf{z_k} - \mu_\mathbf{k})^\mathbf{T} \mathbf{\Sigma_k^{-1}} (\mathbf{z_k} - \mu_\mathbf{k}), \tag{35}$$

where $\mathbf{z_k}$ is a position measurement at some time instant k, $\mu_\mathbf{k}$ is the mean, and $\mathbf{\Sigma_k}$ is the covariance matrix. This formulation, which is an extension of the Euclidean-distance metric, is the definition of the distance between a sample and the mean in $\sqrt{d}$-σ increments. The distance is chi-squared distributed with n degrees of freedom, where n is the dimension of the measurements $\mathbf{z_k}$. The percentage of probability that is required is readily obtained by applying the cumulative-distribution function F of the chi-squared distribution to the Mahalanobis distance; *i.e.*, $p = F(d, n)$, where p is the percentage.

The uncertainty ellipse may now expand and contract based on the total measurement spread. The ellipses are initialized as 99% ellipses, to effectively cover the search region. If the measurements require a much smaller containment region as shown by the calculated Mahalanobis distance for some threshold α of the number of measurements, this new percentage will be used and the ellipse will contract. However, if a great number of are

detected using a similar threshold, then the parameters are relaxed and the ellipse is expanded. The expansion case is illustrated in Figure 21, where previous measurements are tightly clustered, but a recent measurement with a wider spacing that was assigned to the same component causes an expansion of the search region.

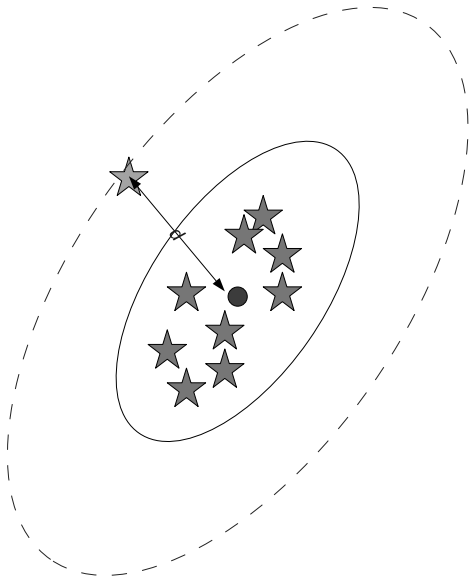


Figure 21. Expansion of the search region as a result of a measurement with a greater spacing.

3.7 Search Region Extraction Using IIP Data

To demonstrate the procedure for extracting search regions from a mixture model, the process is applied to one of the IIP datasets, using one of the mixture models generated in Section 3.5. To be specific, the mixture model generated from the 2011 dataset without the application of search region reduction will be used to demonstrate the process, and show cases where reducing the model can be beneficial.

The first step is to generate the contours of constant probability from the mixture; this is

a straightforward task by iterating over the individual components and applying the requisite equations. The result, when graphed, is shown in Figure 22. From examining this plot, it can be seen that a few components exist where the search region will be of negligible area. These regions can be absorbed into larger regions, if the component merging process does not combine them with other components.

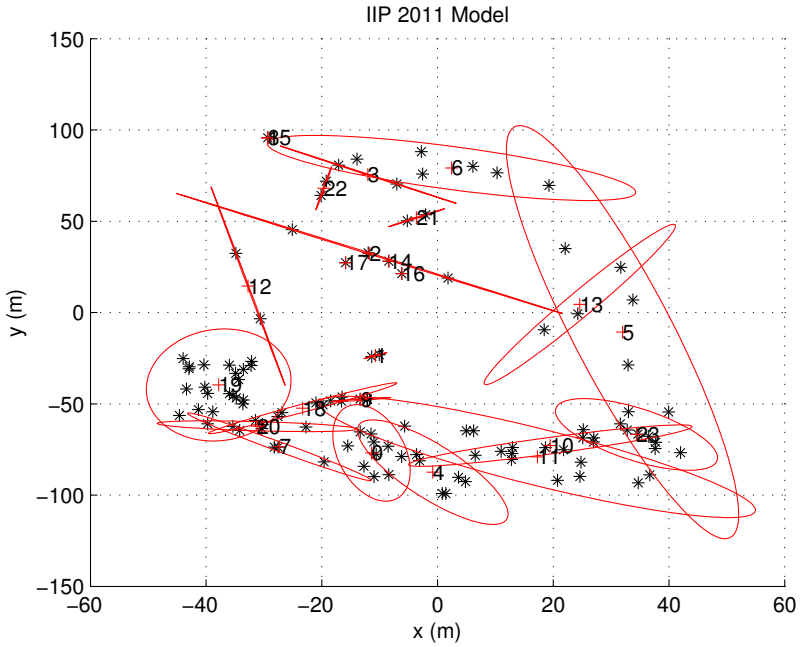


Figure 22. Probability contours for the model generated from the 2011 IIP data set.

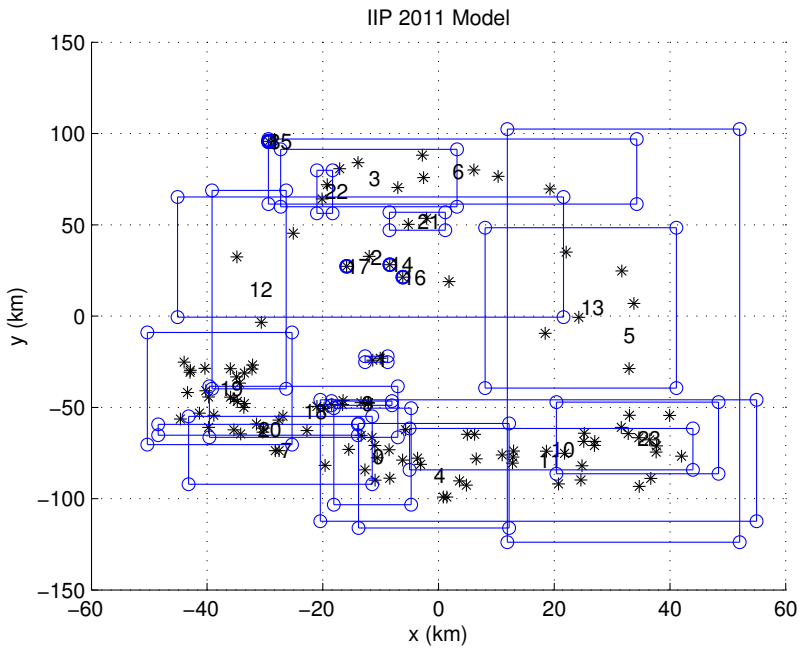


Figure 23. Search regions for the model generated from the 2011 IIP data set.

Once the contours have been generated, the search regions can then be computed, as shown in Figure 23. Once the search regions have been generated, the parameters for the search parameters can be computed. In this case, the parallel-transect and spiral search patterns as described in Section 3.6.3. To compute these parameters, nominal values for the design parameters must be chosen. The sensor field-of-view radius d_s is set to 500 m, and the spacing between points of the spiral ξ is set to 25°. Table 4 shows the results of calculating the parameters. Note that the distances are very large; in an actual mission, the travel distances will likely not be on the order of several kilometers. However, in this case, the distances demonstrate what is required to search regions with areas of this magnitude. Each of the identifiers in the table correspond to the labels in Figure 23.

Most of the resulting lawnmower vertical traversal distances are reasonable; in this case, a reasonable distance is less than 20 km. Additionally, there are regions that have smaller vertical traversals that require many more total traversals to cover the region. This result of having a longer traversal distance with fewer horizontal traversals is more efficient

Table 4. IIP 2011 dataset search pattern parameters.

Component	Lwm. Vertical Dist. D_v	Lwm. Traversal Count N_{trav}	Spiral End Index t_f
0	64.758696	34	527
1	3.997773	11	39
2	80.517556	165	664
3	38.547115	76	314
4	70.151608	65	570
5	277.157686	100	2252
6	43.638355	157	634
7	45.460628	79	370
8	0.818322	2	7
9	2.919672	27	104
10	27.789877	122	488
11	81.424907	186	751
12	133.123829	33	1082
13	107.572144	83	874
14	0.479012	2	4
15	0.818322	2	7
16	0.450948	2	4
17	0.475326	2	4
18	34.220631	81	325
19	75.172304	63	611
20	7.209261	86	344
21	12.058957	25	98
22	28.876854	8	235
23	48.078386	70	391

in covering the region. When compared with the plot of the search regions, there are cases where the sightings taken within these larger search regions are very sparse, and they could have been captured within a different region. Hence, these larger search regions could be removed from the mission altogether, if it is possible to obtain target measurements from smaller search regions.

In the case of the spiral pattern parameters, the magnitude of the final index corresponds directly to the required parameters for the lawnmower pattern. That is, for the larger search regions in terms of area, the index is greater than 500. It should be noted that the total distance traversed within the region, depending on the separation between turns, can be greater than that of the lawnmower pattern.

3.8 Real-time Considerations

When implementing these algorithms on real hardware, resources may be limited as a result of the use of microcontrollers or small embedded processor boards as opposed to a complete computer system. It should also be noted that the overall iceberg observation problem, considering that it uses CMOMMT as a framework, has NP-hard complexity: as a result of the problem being based on CMOMMT, the problem can be reduced to the same underlying problem to which CMOMMT can be reduced. This underlying problem is vertex cover [30].

Algorithm complexity should be also taken into consideration. Expectation-maximization has linear time complexity with regard to the number of measurement dimensions D, the total number of possible iterations I, and the number of measurements N; *i.e.*, $O(NDI)$. Mixture reduction is linear in the number of components K and the measurement dimension D, *i.e.*, $O(KD)$. The clustering algorithm also has a linear time complexity of $O(LD)$, where L is the number of components resulting from the mixture reduction process.

Note that the clustering algorithms are highly parallelizable, and they either may be offloaded to a centralized fuser or computed in pieces across all of the agents. However,

the resulting model must still be reassembled at some central point to be usable. This again suggests the need for the use of a centralized data fuser or the alternative approach of the broadcast of results as they are generated by each of the agents. Once all of the computation is completed, each of the agents individually reconstructs the iceberg model, and then makes decisions based on this reconstructed model.

CHAPTER 4

METHODOLOGY FOR RESOURCE ALLOCATION USING THE ICEBERG MODEL

By using a data-derived model to better characterize the ablation regions on a glacier, icebergs can be efficiently detected before they become a significant issue for arctic operations. However, a default allocation of resources to continuously track these targets may not be the best allocation for detecting new icebergs. An example of such a default allocation would be the division of the region surrounding the glacier into equally-sized rectangular subregions, equal in count to the number of available agents. Such a division is equivalent to a centroidal Voronoi tesselation of the region [57].

This resource allocation issue is a primary reason for developing methods to generate and extract search regions from the model: the new search regions computed from the model should be designed to seek and detect new icebergs as they include previous iceberg positions and speeds.

Therefore, methods for reassigning resources to the regions as produced from the model must be developed. In this chapter, the general assignment problem for assigning agents to tasks is discussed, and a cost function is defined that uses the target model and robot parameters to determine an optimal allocation of resources [58]. Finally, appropriate assignment algorithms for assigning agents to search regions are considered. It should be noted that this assignment approach is one of many possible approaches for allocating agent resources; the primary contribution of this chapter is to show the design methodology for a method using the model as defined in Chapter 3.

4.1 The Assignment Problem

The assignment problem is among the most common of resource allocation problems. The common definition of the problem is as follows: given a set of resources and a set of tasks,

what is the optimal one-to-one assignment of resources to tasks? While this definition seems simple, it may be generalized to any number of dimensions; *e.g.*, developing the optimal assignment of a one-to-many mapping of resources to tasks. For the purposes of this research, however, only the nominal definition of the problem will be considered. As such, a mapping can be represented by entries within a table, which then equates to defining the problem as a two-dimensional assignment.

To formalize the definition of the two-dimensional assignment problem, the following definitions are necessary:

$\mathcal{T}$: A set of N tasks t_j.

$\mathcal{R}$: A set of M resources r_i that can complete a task contained in $\mathcal{T}$.

C: The cost required for a resource r_i to complete a task t_j. The cost function is the one-to-one mapping $C : \mathcal{T} \times \mathcal{R} \longrightarrow \mathbb{R}$. Although it is implied by the name, the cost may not necessarily be monetary or even physical in nature. Energy or additional resources (*e.g.*, tools required to complete a task) may also act as costs.

In two-dimensional assignment problems, an optimal assignment of resources to tasks based on the costs exists that allows for the most efficient method of completing the task. This optimal assignment minimizes the cost across all tasks and resources.

Defining the assignment as the mapping $X : \mathcal{T} \longrightarrow \mathcal{R}$, the objective function of the two-dimensional assignment problem is the following:

$$\min_{\mathcal{R},\mathcal{T}} C, \tag{36}$$

which results in the optimal mapping X.

In the case of the iceberg observation problem, assigning robots to search regions can be considered within the context of the two-dimensional assignment problem: the set of tasks $\mathcal{T}$ is the set of search regions, and the set of resources $\mathcal{R}$ is the set of robots in the

mission.[1]

Many methods and algorithms have been developed over the years to solve the two-dimensional assignment problem. The canonical algorithmic solution to the assignment problem is the Hungarian method (Kuhn-Munkres algorithm) for performing an assignment [59] [60]. This algorithm is a combinatorial method for solving the problem. Of course, other well-known assignment algorithms exist and provide an assignment more efficiently, such as the Jonker-Volgenant shortest-path algorithm [61] and the various auction algorithms by Bertsekas [62].

Solutions to the assignment problem may also use greedy algorithms; greedy, nearest-neighbor assignment [63] is one of the more common ways of implementing such an algorithm. Additionally, decentralized, market-based assignment algorithms lend themselves well to robotics applications. These types of algorithms allow each of the robots to determine their own costs and bid in an auction-like fashion for tasks either tracked by a central arbiter or empirically determined based on the current needs of the robotic system [64–76].

In the case of iceberg observation, since the types of observation tasks are all the same, a standard two-dimensional assignment approach would serve well. The main problem is determining what the cost function should be. Such a function can be defined based on the parameters of the model generated from previous iceberg observations and the physical parameters associated with each of the robotic agents.

4.2 Defining a Cost Function

To determine an appropriate cost function for the iceberg observation problem, an appropriate statement of the assignment problem must be developed. Hence, the assignment problem that is to be solved is as follows:

- The initial allocation of agents has each of the agents assigned to a region derived

[1]For the remainder of the chapter, the terms "two-dimensional assignment problem" and "assignment problem" will be considered as interchangeable.

from the initial mission plan. Each agent has an average speed and a sensor field-of-view that covers a fixed area. Figure 24 illustrates an example of such a default allocation: the mission area is divided into equally-sized, rectangular cells, and agents search within those cells. That is, a centroidal Voronoi tessellation of the region $\mathcal{S}$.

- New regions of varying area and position are then extracted from the computed iceberg model Q using the parameters of the model components q_k, $j = 1...k$. Agents are deployed to the new regions. Figure 25 shows an example of these extracted regions.

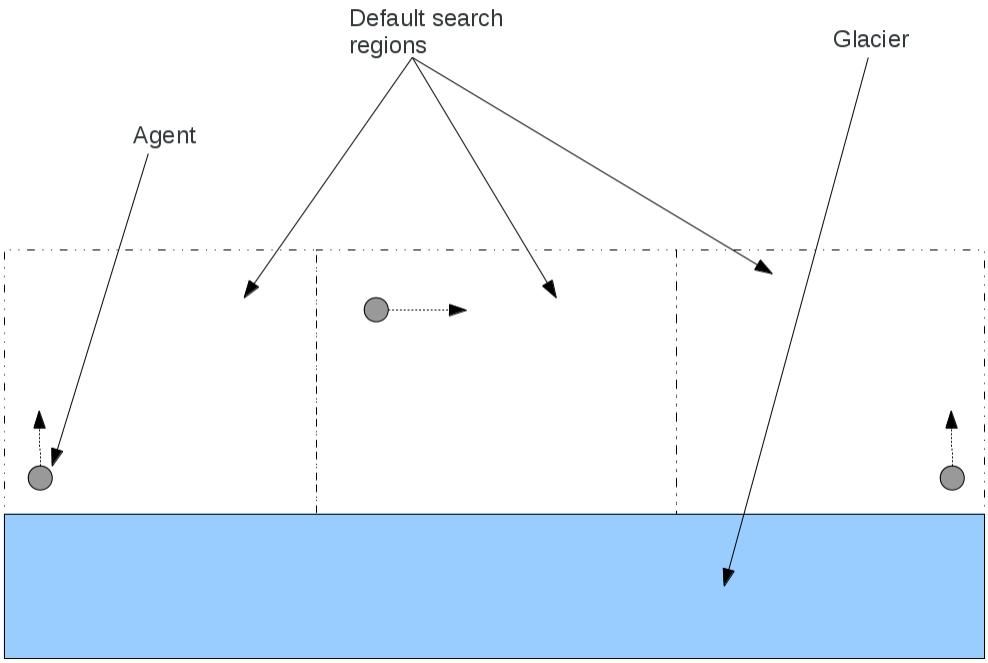


Figure 24. Initial agent allocation.

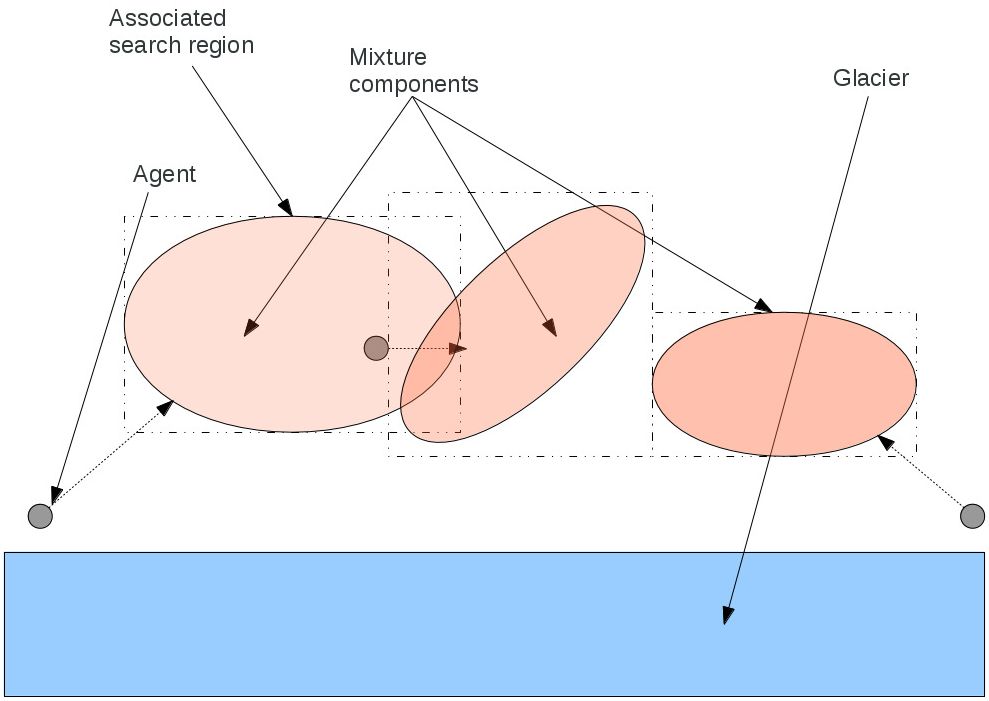


Figure 25. Assignment of agents to search regions based on mixture components.

One of the more common cost functions in the assignment problem uses the Euclidean distance from the position of a robot to a set of given goal points; the assumption is made that the paths that have this distance are both the shortest and, as a consequence, the minimum energy paths to the goal points.

However, in the case of evaluating the cost of a search region for the iceberg coverage problem, additional factors must be considered. Such factors include how quickly and efficiently the region can be covered while still acquiring a significant number of the targets contained within it. To accommodate these factors, two weighting factors are placed on the distance to the centers of the search regions, which will generally be defined by the means of the Gaussian mixture components used to generate them.

This leads to a definition of the cost function C as follows:

$$C = \frac{A_R}{A_{FOV}} \frac{\bar{v}_j}{v_r} \sqrt{\left(\mathbf{x} - \mu_j\right)^T \left(\mathbf{x} - \mu_j\right)}, \tag{37}$$

Table 5. Cost function parameters.

Variable	Definition
A_R	Area of a given search region.
A_{FOV}	Area of a given sensor field of view.
$\bar{v}_j$	Average velocity of targets within a search region j.
v_r	Average velocity of a given robot.
$\mathbf{x}$	Robot position.
μ_j	Component mean or center of a given search region.

where the symbols are given in Table 5 and $j = 1...k$, k being the number of components in the model Q.

The first weighting factor A_R/A_{FOV} depends on the area of a search region and the field of view of the robot's sensors, which is the contribution of $\mathcal{R}$ to the cost function, where $\mathcal{R}$ was defined in Section 4.1. This factor, as defined, is the ratio of the search region's area to the area covered by the robot's sensor field of view. The justification is that this ratio will be 1 when the areas are equal, indicating that the only relevant cost is the distance required to travel to the sensor. However, the cost will increase if the search area increases without increasing the area that the agent's sensor can cover; *i.e.*, more energy is required to cover the search region. The converse is true for smaller search areas.

The second weighting factor $\bar{v}_j/v_r$ relies on the average velocities of targets that have been acquired and the average robot velocity; it is the contribution of $\mathcal{T}$ to the cost function, where $\mathcal{T}$ was defined in Section 4.1, in addition to the distance to the search region center. This factor implies that if these quantities are equal, then again, no additional energy is required by the robot to have parity with the speed of the targets. However, for higher target velocities with respect to the robot, the cost will increase as the robot will have to increase its speed to acquire targets, and likely consume more energy to do so.

Table 6. Parameters used for the cost function sweep.

Variable	Value
A_R	100 m^2
A_{FOV}	4π m^2
$\bar{v}_j$	0.1 m/s
v_r	100 m/s

Figures 26 and 27 are examples of holding a set of the cost function parameters constant and sweeping one of the other parameters to demonstrate the effects on the cost function. In all cases, the distance to the region has been held fixed at 100 m. The other fixed parameter values are given in Table 6.

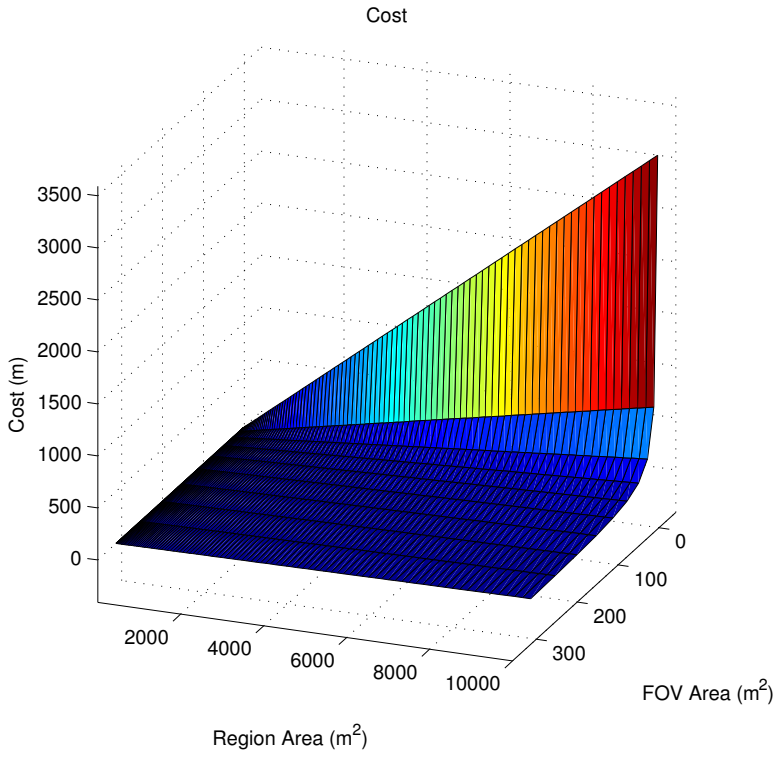


Figure 26. Plot of cost C versus versus FOV area A_{FOV} and search region area A_R. The cost behavior is linear with respect to the region area, and exponential with respect to the FOV area.

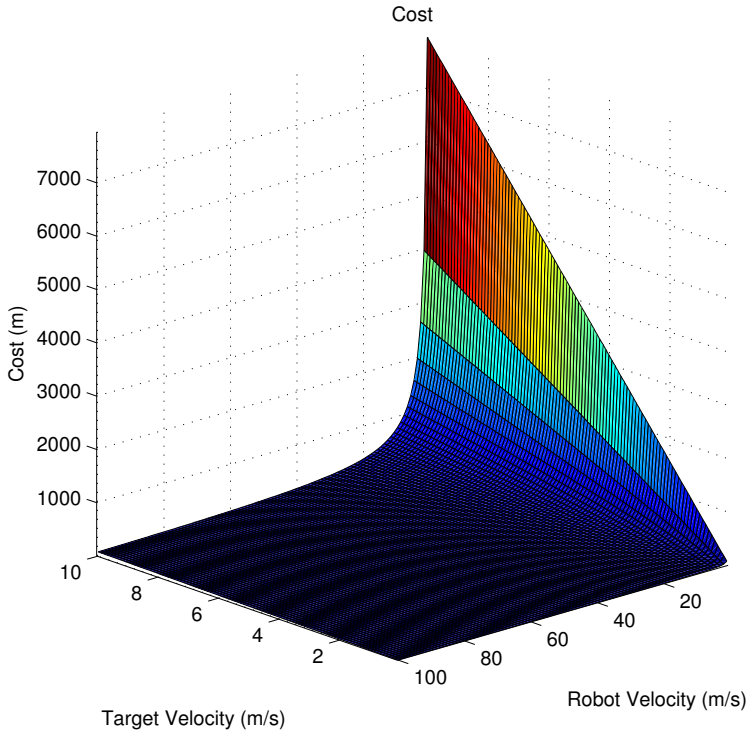


Figure 27. Plot of cost C versus target velocity $\bar{v}_j$ and average agent speed v_r. The cost behavior is linear with respect to the target velocity, and exponential with respect to the robot velocity.

These plots show that, as one would expect from the equation for the cost function, that the overall behavior of the cost is linear with respect to the region parameters and exponential with respect to the agent parameters. Note that while search region area can grow to be extremely large, as demonstrated by the study of the IIP database data in Section 3.5, the velocities of these targets remain relatively low. Hence, for search regions, the rise in cost as the region size increases will have a steeper slope than that of the targets, as demonstrated in Figure 26.

Similarly, depending on the sensors and agents used, widening the sensor field-of-view would likely be cheaper in terms of energy than it would be to increase the agent velocity to more quickly cover an area. Therefore, like the case of search regions and target velocity, the cost will decay more sharply as the sensors are improved, which is shown in Figure 26.

4.3 Selecting an Appropriate Assignment Algorithm

4.3.1 Motivation

The selection of an algorithm to use depends on how optimal the resulting assignment solution should be in comparison to the computational resources required. The consideration in this particular case is whether the best allocation for the agents is the most resource-conservative when moving from one search region to another. It is of little use to provide an optimal allocation if the agent does not have enough energy to conduct a search. For the given cost function, this can occur for high agent velocities coupled with small search regions in comparison to the sensor field-of-view, without verifying the resources required to operate the sensor into consideration.

Therefore, the relative benefits of a sub-optimal allocation must be taken into consideration. First, note that cost minimization is the problem of interest. A greedy approach will always select the smallest cost first, regardless of the considerations that were a part of its computation. This will occur in the case of high agent velocity coupled with small search regions. Hence, this suggests that an optimal approach, with appropriate computation times, would be a better choice for the assignment algorithm that lies at the core of the resource allocation algorithm.

4.3.2 Auction Algorithms

Based on the requirements for the complete resource allocation algorithm for this application, the Bertsekas forward-reverse-auction algorithm [62] will be used. The algorithm is an extension of the standard auction algorithm, which, by its name, resembles an auction process in that for a given object (*e.g.*, a task), the actor that wants the object attempts to make it as unattractive as possible to the other actors in the auction such that the desiring actor wins the object once the auction process has concluded. In particular, the actor makes an object undesirable by taking turns raising the price at which the object will be sold. The algorithms will be briefly summarized here; further details, theorems, and proofs associated with these algorithms may be found within the reference [62].

The forward-auction process operates in this manner; the actors attempt to maximize the value of their desired objects with respect to the price, while raising the price during each round of the auction. Each round ends with a partial assignment of objects to actors; several rounds of auction are conducted until each actor has an object assigned to it.

Complementary to forward-auction is the reverse-auction process. The main difference between the two auction processes is what is assigned to whom: for forward-auction, actors are assigned to objects; for reverse auction, objects are effectively assigned to actors. Instead of maximizing an individual object value, the profit associated with winning the object is maximized by the object for the assigned actor.

Before defining the algorithms, some definitions in addition to the definitions given in Section 4.1 are required. For these definitions, the actors are the robots contained in the set $\mathcal{R}$, and the objects are the tasks contained in the set $\mathcal{T}$.

The definitions for the forward-auction algorithm are as follows:

P: The vector of prices p_j of each of the tasks contained in $\mathcal{T}$.

V: The value v_j of each of the tasks contained in $\mathcal{T}$ with respect to its corresponding entry p_j in the price vector P.

$\mathcal{B}_i$: The vector of bids $b_{i,j}$ for each of tasks t_j by the robots r_i. Each robot has its own vector of bids.

Conversely, the definitions for the reverse-auction are as follows:

Π: The vector of the profits π_i for each of the robots contained in $\mathcal{R}$. The difference between profits and prices is that a profit is the additional value received by a robot for winning a task, as opposed to obtaining it at the highest price.

W: The value w_i provided by each of the robots contained in $\mathcal{R}$ with respect to its corresponding entry π_j in the profit vector Π.

$\mathcal{B}_j$: The vector of bids $b_{j,i}$ for each of robots r_i by the tasks t_j. Each task has its own vector of bids.

Both the forward-auction and reverse-auction algorithms have two main steps: a bidding and an assignment step. The forward-auction algorithm, summarized, proceeds as follows:

Bidding step: For each of the tasks t_j, using the values of C for each r_i and t_j contained within the cost matrix (*i.e.*, c_{ij}), determine which task t_j has the best value v_j (*i.e.*, the maximum value) in the set of all tasks:

$$t_j = \arg\max_{t_j \in \mathcal{T}} \left\{ c_{ij} - p_j \right\}. \tag{38}$$

The value that corresponds to this task is

$$v_j = \max_{t_j \in \mathcal{T}} \left\{ c_{ij} - p_j \right\}. \tag{39}$$

The next best task value is then obtained:

$$u_j = \max_{t_j \in \mathcal{T}, t_j \neq t_{j,next}} \left\{ c_{ij} - p_j \right\}, \tag{40}$$

where $t_{j,next}$ is the task that possesses this next best value.

Finally, the given bid for a particular robot r_i is computed using the following equation:

$$b_{i,j} = v_j - \left(c_{i,j} - p_j \right) - 2u_j + 2\epsilon, \tag{41}$$

where ϵ is a small value, bounded above in this study by $\epsilon = 10^{-9}$. The bidding step is repeated until all of the robots that have not been assigned to a task have issued a bid for any of the remaining tasks.

Assignment step: For each of the tasks that received a bid, given the set of robots that bid on a particular task, raise the price of that task to the maximum bid:

$$p_j = \max_{t_j \in \mathcal{T}} b_{i,j} \tag{42}$$

Remove any previous assignments to those tasks, and assign tasks to the robot which has currently won that task. The algorithm terminates once all of the tasks have been assigned to a robot.

Given the description of the forward-auction algorithm, reverse-auction may then be described as follows:

Bidding step: For each of the robots r_i, using the values of C for each r_i and t_j contained within the cost matrix (*i.e.*, c_{ij}), determine which robot r_i has the best value w_i (*i.e.*, the maximum value) in the set of all robots:

$$r_i = \arg\max_{r_i \in \mathcal{R}} \left\{ c_{ij} - \pi_i \right\}. \tag{43}$$

The value that corresponds to this robot is

$$w_i = \max_{r_i \in \mathcal{R}} \left\{ c_{ij} - \pi_i \right\}. \tag{44}$$

The next best robot value is then obtained:

$$u_i = \max_{r_i \in \mathcal{R}, r_i \neq r_{i,next}} \left\{ c_{ij} - \pi_i \right\}, \tag{45}$$

where $r_{i,next}$ is the robot that possesses this next best value.

Finally, the given bid for a particular task t_j is computed using the following equation:

$$b_{i,j} = w_i - \left(c_{i,j} - \pi_j \right) - 2u_i + 2\epsilon, \tag{46}$$

where ϵ is a small value, bounded above in this study by $\epsilon = 10^{-9}$, as in forward auction. The bidding step is repeated until all of the tasks that have not been assigned to a robot have issued a bid for any of the remaining robots.

Assignment step: For each of the robots that received a bid, given the set of tasks that bid on a particular robot, raise the profit of that robot to the maximum bid:

$$\pi_i = \max_{r_i \in \mathcal{R}} b_{i,j} \tag{47}$$

Remove any previous assignments to those robots, and assign robots to the task which has currently won that robot. The algorithm terminates once all of the robots have been assigned to a task.

Forward-reverse-auction is a combination of the two algorithms, where rounds of forward-auction are alternated with rounds of reverse-auction. Both price and profit vectors are maintained each iteration in addition to the current partial assignment of robots to tasks. Ultimately, the forward-reverse-auction algorithm converges to the optimal assignment in much less time than either of the algorithms operating alone.

In the forward-reverse-auction algorithm, potentially multiple rounds of forward-auction are run. At the conclusion of the assignment step, the profit vector Π is set according to who won the current assignment and the current prices P of the tasks. Similarly, multiple rounds of reverse auction are run, and once the assignment has been made, the price vector P is modified according to the current assignment Q and the current profit vector Π for each task. This back-and-forth approach between each of the algorithms assists in converging to the solution faster and prevents price wars. A price war is defined as a contention between two bidders where each bidder raises the price by a small increment during each iteration of the bidding step [62]. Price wars most commonly occur when using the forward-auction algorithm. This aspect of the forward-reverse auction algorithm, a reduction in price wars, is a highly desirable one for this particular resource allocation problem: when the model has been shown to reflect a change in iceberg behavior such that new search regions must be generated and the agents must be reassigned, the assignment portion will not require the bulk of the processing time with respect to the time required to initially generate the model, communicate the changes in target assignment, and move the agents from their current search regions to their new assignments.

4.4 Arbitration of Resource Allocation

Once the costs have been computed with respect to the search regions, arbitration is employed to execute the assignment algorithm and assign the agents to the search regions. Arbitration is officially defined as the process by which a dispute between two parties is settled by an impartial third party. In the case of resource allocation, this definition may be modified to state that arbitration is the process by which resources are allocated to agents by another agent. In this case, while the assignment process is ultimately deterministic, arbitration is used to ensure that the model that is used for reassignment is consistent across all of the agents in the mission; *i.e.*, other agents may have collected measurements, but have yet to transmit them, yet they have already been incorporated into their own individual versions of the iceberg model. The fact that an assignment algorithm will be used for processing the cost matrix and performing region assignment will play a key role in determining how arbitration will be handled.

Several methods exist for handling the arbitration of a resource allocation algorithm. A few examples are as follows, representing generalizations of most of the methods used in resource allocation:

- A single arbiter separate from the agents.
- An agent that acts as an arbiter.
- Arbitration is distributed across agents.

There are advantages and disadvantages to each of these approaches. For the first option, the single arbiter is a kind of base station for the operation of the agents. Therefore, reliable communications are required to ensure that it can provide information to the agents in a timely fashion. Since the world is not ideal, the communications channel would certainly not be completely reliable. The arbiter might be required to be placed far away from the agents, such as being based on an oil platform many kilometers away. An agent acting as the arbiter is a better option, as it will be closer to its fellow agents and communications

will be more reliable. However, if the agent must drop out of the mission for any reason, the advantages of a single fixed arbiter are lost.

The last option thus sounds like the best option: communication between agents will have theoretically fewer errors as a result of proximity, and the agents are handing off the arbiter role, ensuring that an agent dropping out of the mission will not hinder the rest of the agents. Much like the individual methods of arbitration, there are multiple ways of implementing distributed arbitration:

- A single agent acts as arbiter until it is forced to hand off arbitration to another agent.
- The agents periodically call for a vote on which agent should be the arbiter.
- The agents self-arbitrate by determining allocations individually and collectively fusing them together to produce the final allocation.

Again, there are advantages and disadvantages to each of these options. For the last two options, error correction or fail-safe procedures must be in place to ensure that if the agent fails while tallying votes or producing the final allocation, the other agents can recover. For example, if the election method is considered, the agents do not receive a response in a certain amount of time, a new election is held, with a new vote counter determined by a hand-off algorithm, such as a round-robin scheduling algorithm. Alternatively, the votes can be broadcast to all agents, they are all counted individually, and the winner declares itself the arbiter, which is a variant of the third option. However, communications errors could cause votes to not be received by certain agents.

Hence, the option that will be chosen is the first option, as it provides sufficient reliability at the sacrifice of a single agent having to store a significant amount of the state required to arbitrate the rest of the agents. It should also be noted that even if the agent is unable to perform the hand-off in time, some fail-safe measures can be put into place. If the other agents determine that the model has changed sufficiently that a reassignment should be occurring (as they all hold a global form of the model), the agents can elect a

new arbiter, or an agent automatically assumes arbitration duties based on a fixed sequence of agents.

4.5 Algorithms for Region Assignment

Given the definitions and the selected algorithms in the previous sections, a complete resource allocation algorithm may now be defined. Note that prior to this assignment process, if a model has not been computed, the agents are assigned to the default search regions. First, some initial conditions must be stated with regard to the system:

- The default resource allocation is the division of the region around the glacier that is currently calving icebergs into equally sized search regions. The number of initial search regions is equal to the number of agents in the system.
- A default arbiter is assigned at the start of the mission, but it is not assumed to be the same arbiter throughout the mission, depending on agent dropout.
- A model is generated prior to assigning agents to search regions.
- The search regions are generated from the model prior to assignment.
- An assignment is not initiated until the arbiter has received appropriate state vectors from each of the agents in the mission.

The steps in the search region assignment algorithm for the global iceberg model as run on the current arbiter are thus as follows:

1. The model is first verified as a valid model (*i.e.*, there are no statistical validity issues). If it is not, then the system exits from the assignment routine, and the agents remain in their currently assigned search regions.
2. The number of measurements obtained for the model are verified as sufficient for providing an accurate picture of iceberg activity. If not, then the system exits from

the routine, and the agents remain in their currently assigned search regions. Alternatively, this test can be bypassed if one of the agents informs the arbiter that an assignment must take place as a result of some significant event (*e.g.*, an ablation region dropped out of the model in an agent's local model).

3. The model's components are computed, and then search regions are computed from the components. Combined with the current state and sensor properties of each of the agents in the mission, the forward-reverse-auction assignment algorithm is executed.

4. The assignment vector resulting from the forward-reverse-auction assignment algorithm is examined. If an agent was assigned to a particular search region, a redirect request is sent over the network to that agent.

Note that the final step suggests that partial assignments can occur (*i.e.*, there are more agents in the mission than there are search regions). In this case, agents either revert to the default resource allocation, otherwise they remain at the search region to which they were last assigned. In addition, there is the issue of *region hand-off*: one agent may be placed in a similar or identical region as another agent. Note that the regions will generally not be identical: this would only occur as the result of a forced reallocation and no new iceberg position measurements having been obtained prior to this reallocation. The impact of hand-off depends on the assignment algorithm that is used: so long as a unique allocation where agents are placed in unique and separate regions is produced each time that the allocation algorithm is run, region hand-off is a non-issue. This is the case in the reallocation algorithm described here. If the algorithm can produce such resource conflicts, then the current and previous assignment vectors must be compared to ensure that there will not be a conflict. If a conflict exists, then either the assignment must be run again, or if there are more regions than agents, the conflicting resource must be allocated to one of these "extra" regions. Alternatively, if the search region has an "excessively large" area, the region can be tesselated into smaller regions, and the resource allocation algorithm run once

more against this new set of regions. The constraint on reverting to the default allocation is whether the agent can travel the distance to the center point of their default search region without wasting too much energy to do so.

Finally, once conditions for hand-off arise (*e.g.*, the arbiter must exit the mission as a result of low energy levels), arbiter hand-off is accomplished using the following algorithm based on the reasoning in Section 4.4:

1. The current arbiter searches for a candidate to act as the new arbiter. The selection process starts by examining the list of agents that have most recently reported their positions to the arbiter for assignment to search regions. An agent is first selected from this list as an arbiter candidate, based on current loading.

2. The current arbiter queries the agent via network message to determine if it can become the arbiter. The agent issues a reply based on certain conditions of its state; *e.g.*, how close is it to dropping out of the mission as a result of energy levels.

3. If the agent cannot become the arbiter, the current arbiter moves down the list of agents until the list of agents is either exhausted or an agent states that it can become the arbiter. If the list is exhausted, the current arbiter selects the arbiter candidate to be the best fit in terms of remaining energy.

4. The current arbiter issues the arbiter hand-off network message to the candidate.

5. Upon receipt of the message, the arbiter candidate broadcasts a message to all agents that it is the new arbiter. All agents in the mission then adjust their internal state to reflect that a new arbiter has been selected, including the original arbiter. Current agent states are then sent to this new arbiter as confirmation.

6. If the original arbiter does not receive the new arbiter message within a certain time-out, then it attempts to notify the candidate again. After a fixed number of attempts, the original arbiter chooses a new candidate using the same method as above, and the

steps repeat until a new arbiter has been selected. If no candidates report, then the original arbiter starts selection again from the start of the list.

Note that this algorithm has a potential flaw, which is difficult to eliminate using a distributed arbiter approach: the condition may exist that the original arbiter is never able to select a new arbiter. Such a condition may occur as a result of network errors: the original arbiter never receives replies, or the arbiter's original transmissions never reach the arbiter candidates. The arbiter would become stuck in an loop as it continues to repeat the hand-off process. The issue would manifest itself in search region reassignments never taking place. Such a loop can be broken by having the arbiter defer to a base station when such an issue arises; *e.g.*, if the hand-off process repeats N amount of times with no success, where N is a chosen limit on attempts, inform the base station that it must handle arbitration. The original arbiter can then freely leave the mission area.

CHAPTER 5

EXPERIMENTAL SETUP

In Chapter 3, the methodology for computing models of ablating target sources and metrics associated with those models was outlined, applied to a problem definition that was obtained by extending the CMOMMT problem definition. Models were generated from the iceberg sighting databases provided by the IIP to demonstrate how the modeling techniques can extract the structure from the target measurements, and how the metrics for each of the years compare with each other. Chapter 4 described a methodology for assigning resources to monitor those target sources by using a distributed set of agents that communicate with each other to determine to which search regions they should be assigned, based on a cost function that is computed from the metrics obtained from the models in Chapter 3 and the properties of the individual agents, including their sensors.

In this chapter, the requirements, assumptions, and implementation of the robot controller used to control individual agents are provided. In addition, the simulation platforms and robot hardware platform are described.

5.1 Implementation of the Robot Controller

The techniques and algorithms described in Chapters 3 and 4 may be integrated into a complete controller for a mobile robot acting as a single agent in the iceberg observation system. In this section, the configuration and the algorithmic components of the robot controller are described.

5.1.1 Assumptions

A few assumptions must be made with regard to the robot controller and the robot itself, as the primary concern in this section is how the algorithms given in previous chapters are implemented, rather than some of the low-level details that are generally part of any advanced robotic system. To emphasize, these algorithms may be implemented on any

type of robot that is required for monitoring such ablating target-sources; specific platforms such as autonomous underwater vehicles (AUVs) or unmanned aerial vehicles (UAVs) are not assumed in this implementation.

The robot is assumed to have an obstacle avoidance capability that is integrated into its control laws. Examples of such control laws are the algorithms given in [77], which provide a fuzzy-logic approach to behavior integration.

The robot is also assumed to have a localization method. Examples of localization methods that could be used on a robot operating in similar conditions include dead-reckoning via odometry, satellite-based navigation such as the Global Positioning System (GPS), inertial navigation, or a fusion of various localization techniques [78] [79]. It is likely that odometry will have a lower emphasis in a localization solution for this problem, as a result of the difficulty in obtaining odometry with the types of autonomous vehicles that would likely be used in such an observation mission. Hence, the primary means of localization would likely be an integrated inertial measurement unit (IMU) and GPS solution [80].

The primary means of controlling robot movement is assumed to be manipulation of the linear (v) and angular (ω) velocities of the robot. Note that the cost function in Section 4.2 requires the assumption that an *average speed* of the agent can be computed; such an average can be computed from the velocities used to control the robot.

The robot must also have a means of communicating with other agents, and potentially with a base station or other communications stations for sharing the results of the observation mission. For a lab-based implementation, such a means of communication would include Wi-Fi-based or Zigbee-based communications, which is what will be used in the experiments. In the field, inter-agent radio links or satellite links would provide communications across agents.

Finally, the robot must have sensors for obtaining target measurements. While the modeling methodology in Chapter 3 assumes a world-relative Cartesian measurement model,

such as an Earth-Centered, Earth-Fixed (ECEF) Cartesian frame, no requirement is imposed on the sensor to provide measurements in this coordinate frame. Indeed, a chain of coordinate transforms may be required to transform from sensor coordinates to the desired frame. Such coordinate transforms may be found in any reference on navigation; [80] provides several of the more important coordinate transforms, such as transforming to ECEF or local geodetic (*e.g.*, East-North-Up) coordinates from and to latitude, longitude, and altitude. If the sensor provides range-bearing-elevation measurements, then the first transformation is the well-known spherical-to-Cartesian transformation, with an offset to the bearing angle to transform it into an azimuth angle.[1] Also, as it is a key assumption given in Section 3.1.2, the sensors are assumed to be noisy. Observations are assured not to be perfect.

5.1.2 Controller Organization

The controller is organized into a set of loosely-defined components, each with its own particular function. The primary component of the robot controller is referred to as the *main component*, which handles reading the current state of the robot hardware, simulated or real, and executes much of the controller logic, but there are other controller components that each handle a specific task.

A simplified overview diagram of the organization and function of each of the controller components and their internal modules is shown in Figure 28. Arrows indicate the direction of outputs from each module to another; each component runs as a loop, such that the topmost module in each component generates the data that is operated upon by the modules further down.

[1] The bearing angle in most sensors is referenced from zero degrees true-north, which would be 90° azimuth, if referenced from the x-axis.

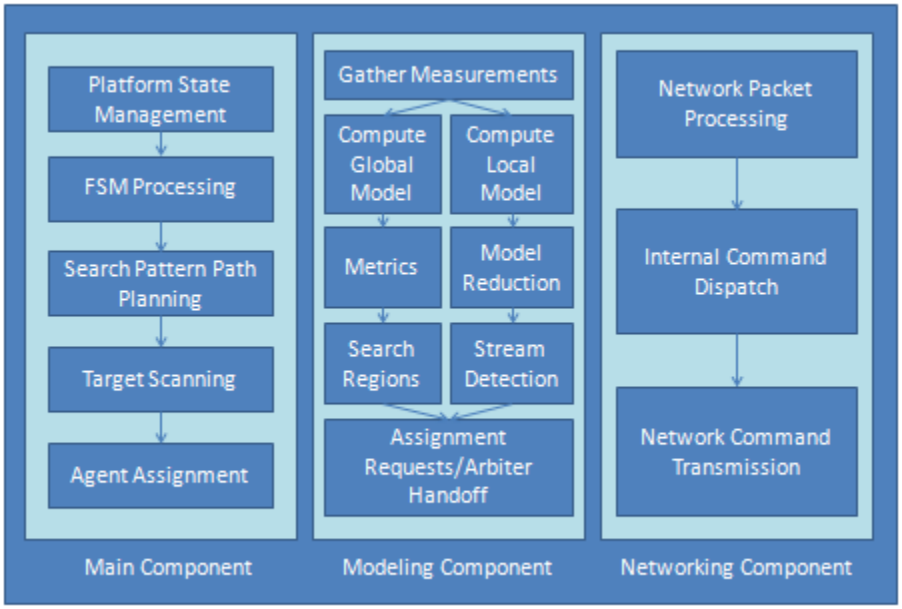


Figure 28. Organization of controller components and their functionality. Although it is not shown in this illustration, the three components communicate as necessary to pass data.

A more detailed description of these components, including the main component, is as follows:

- *Main component*: The main component handles acquiring target measurements, maintaining robot state (position, velocity, etc.), path planning in terms of search patterns, and executing individual search patterns. The robot's finite state machine (FSM) is contained within the main component, which is further explained in Section 5.1.4. This component runs in a continuous loop, executing each of its individual tasks. This loop is designed to run each cycle in the same amount of time; this time constraint exists to ensure that the main component does not desynchronize with respect to any timing requirements imposed by the hardware platform on which it is run, since it is actively querying the underlying platform, unlike the other controller components. For this implementation, the execution rate of the main component is 25 times per second (a period of 40 milliseconds), locked to the execution rate of the

underlying driver software.

- *Networking component*: The networking component processes user datagram protocol (UDP) packets received from other agents. These packets control resource allocation, measurement sharing, and more basic operations such as current behavioral state. This component is effectively a large C++ `case` statement that examines the header of every received packet and executes the appropriate logic for each type of packet, dispatching commands to the appropriate component.

- *Modeling component*: The modeling component recomputes the models based on measurements acquired by the agent and transmitted to the agent over the network. The models are computed as described in Chapter 3. Both the local and global models are computed, including their associated metrics. This component also handles component reduction and target-stream detection.

All three components are activated at controller initialization. The networking and modeling components wait to receive data as a result of main component processing before executing. The main component waits until fresh state data is received from the robot prior to executing the next iteration of the main loop.

5.1.3 Search Pattern Controllers

Each agent is capable of executing a search pattern from a bank of search patterns as part of a scanning mission. For this implementation, three search patterns have been defined; two of which have been previously discussed in some detail with regard to their mathematical definitions in Section 3.6.3. Figure 29 is an illustration of the three search patterns. From left to right in the illustration, they are as follows: the patrolling pattern, the parallel-transect/lawnmower pattern, and the arithmetic/Archimedean spiral pattern. In this section, the discussion of the search patterns will be qualitative rather than quantitative in nature.

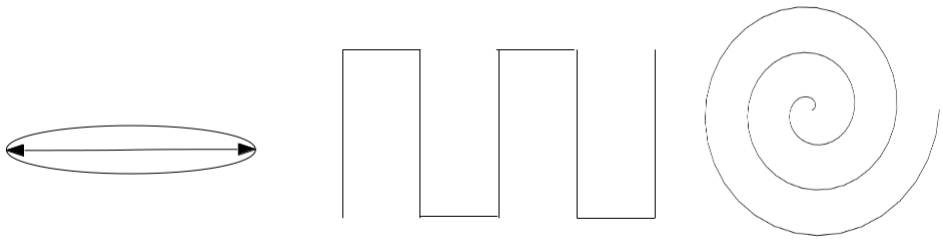


Figure 29. Search patterns used by the controller. From left to right: patroller, parallel-transect/lawnmower, and arithmetic spiral.

The search pattern controllers built into the controller are as follows:

- *Patrolling pattern*: The behavior embodied in this search pattern is similar to what is often referred to as a "loitering" pattern [81]. Loitering patterns are search patterns where the agent travels in a closed loop around a particular area; this loop is often circular. The patrolling pattern is a pattern where the robot travels in a straight line between two points; *i.e.*, effectively a compressed loitering pattern. This search pattern is intended for computing an *a-priori* distribution to assist in determining the initial allocation of agents to search regions; *e.g.*, running several patrol scans with one agent and then switching off to a different search pattern in concert with the other agents included in the mission.

- *Parallel-transect/lawnmower pattern*: This pattern is the most common pattern used in search operations, as a result of its coverage properties [82]. The result resembles the pattern formed when mowing a lawn, hence the name. For rectangular regions, it provides the best coverage. Modifications to this pattern have been made to provide for more efficient sampling of a region [41], but for the controller implemented for the iceberg observation problem, the standard lawnmower pattern will be used.

- *Arithmetic/Archimedean spiral pattern*: This pattern is another commonly used pattern; it provides good coverage for regions that are more circular than rectangular in

shape. The name "arithmetic spiral" comes from the fact that each of the turns of the spiral are equally spaced, as opposed to the logarithmic spiral, where the spacing between turns increases based on a logarithmic progression [83]. The reference to Archimedes originates from the fact that during his studies of geometry, he elucidated the geometric properties of the arithmetic spiral. The property of the spiral that the turns are equally spaced allows for the arithmetic spiral to act as an analog to the lawnmower pattern for circular regions.

5.1.4 Agent Finite State Machine

A finite state machine (FSM) approach was used to implement the robot behaviors in the main component; specifically, the FSM was implemented as the core part of the main processing loop that runs each time a new robot state is received from the underlying robot driver software. Each state in the state machine uses a separate controller for a particular robot behavior.

A state-transition diagram of the state machine used for this implementation is shown in Figure 30. Note that this is a state machine that may be used as part of any robot controller that must perform the task of retrieving data from a goal point.

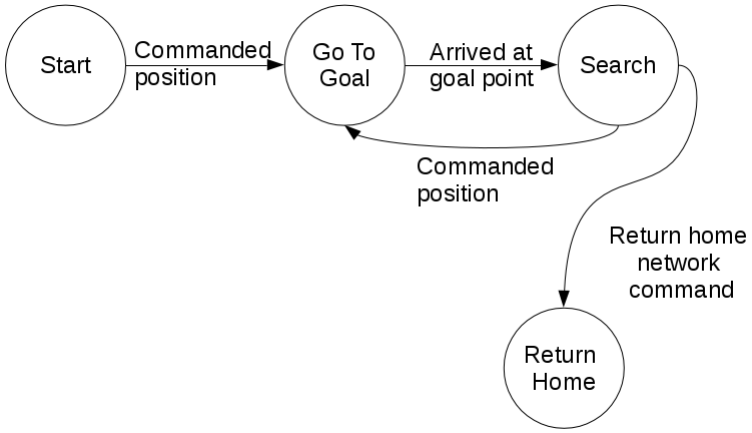


Figure 30. Finite state machine for the robot main loop.

5.1.4.1 Start State

The *start* state is the initialization state for the controller; the FSM never returns to this state during execution. The controller loads a mission plan as part of initialization. This mission plan contains information on all of the agents that are present in the mission. Such information includes the fixed agent names, the network information required for communication between agents and the software drivers controlling the agents, and indication of which agent is the starting arbiter for search-region assignment.

Additionally, a search pattern is configured based on parameters provided to the controller. These parameters include the number of traversals, the length of a single traversal, and the distance between individual sub-points within a traversal. The robot then transitions to the *go-to-goal* state.

5.1.4.2 Go-to-goal State

While the robot controller is in the *go-to-goal* state, the robot is advancing to a commanded position. The only commanded position to which the robot would be moving is the center point of a newly assigned search region.

The controller uses "follow-the-carrot" navigation to move between points. "Follow-the-carrot" navigation may be explained as follows. A checkpoint (x_c, y_c) is computed and placed a short distance ahead of the robot, given the robot's pose (x_r, y_r, ϕ_r). This checkpoint lies between the current known position of the robot and the goal point (x_g, y_g) to which it is moving. By the use of proportional feedback-controllers, the robot controller computes the appropriate speed v and angular velocity ω to move and turn toward the checkpoint. As the robot reaches these checkpoints, the robot controller continues to generate and drive toward new checkpoints until it reaches the goal point. This means of control provides a stable method of moving from a starting point to an ending point. Algorithm 1 illustrates the complete navigation algorithm in pseudocode. Note that a procedure is required to calculate the velocity to move and steer toward a goal point from a given starting point; Algorithm 2 is the form of the velocity calculation algorithm that is used in the

implementation of the robot controller. Various angular corrections have been omitted from the pseudocode listing for clarity. Additionally, the atan2() function is the four-quadrant arctangent function, a common component of most mathematical software libraries. The velocity calculation algorithm allows for reconfiguration of how the velocities are computed based on particular robot properties; in this case, the maximum and minimum linear (v_{min} and v_{max}) and angular (ω_{min} and ω_{max}) velocities that a robot platform is capable of.

Algorithm 1 "Follow-the-carrot" navigation algorithm.

Require: (x_r, y_r, ϕ_r) set to the current robot pose; (x_g, y_g) set to the goal position; a switching range set to $s > 0$; a limit N set to the number of checkpoints; and a maximum distance between the robot position and a checkpoint set to $D_{max} > 0$.

Ensure: v and ω will steer the robot toward the goal.

1: **procedure** FollowCarrot(x_r, y_r, ϕ_r, x_g, y_g, s, N, D_{max}, v, ω)
2: $d \leftarrow \sqrt{(x_g - x_r)^2 + (y_g - y_r)^2}$ ▷ Calculate distance to goal.
3: **if** $d \geq s$ **then** ▷ Compute checkpoint; initial checkpoint is the goal position.
4: $x_c \leftarrow x_g$
5: $y_c \leftarrow y_g$
6: **for** $i \leftarrow 1, N$ **do**
7: $d \leftarrow \sqrt{(x_c - x_r)^2 + (x_g - x_r)^2}$
8: **if** $d \leq D_{max}$ **then**
9: **exit for** ▷ A proper checkpoint has been calculated.
10: **end if**
11: $x_c \leftarrow x_r + \frac{x_c - x_r}{2}$
12: $y_c \leftarrow y_r + \frac{y_c - y_r}{2}$
13: **end for**
14: CalcVelocities(x_r, y_r, ϕ_r, x_c, y_c, v, ω)
15: **else** ▷ Arrived at goal.
16: $v \leftarrow 0$
17: $\omega \leftarrow 0$
18: **end if**
19: **end procedure**

Algorithm 2 An implementation of the velocity calculation procedure.

Require: W is set to an appropriate weighting factor.

procedure CALCVELOCITIES(x_r, y_r, ϕ_r, x_g, y_g, v, ω)
    $\Delta x \leftarrow x_g - x_r$
    $\Delta y \leftarrow y_g - y_r$
    $d \leftarrow \sqrt{\Delta x^2 + \Delta y^2}$
    $\phi_{new} \leftarrow \text{atan2}\,(\Delta y, \Delta x)$
    $\Delta\phi \leftarrow \phi_{new} - \phi_r$
    $v \leftarrow v_{min} + \frac{d}{W}\,(v_{max} - v_{min})$
    $\omega \leftarrow \omega_{min} + \frac{|\Delta\phi|}{\pi}\,(\omega_{max} - \omega_{min})$
    if $\Delta\phi < 0$ **then**
        $\omega \leftarrow -\omega$
    end if
end procedure

Once the robot reaches its set goal point, the state machine transitions to the *search* state.

5.1.4.3 Search State

Upon entering the *search* state, the robot begins to execute its search pattern, as set by the initial configuration. The robot remains in the *search* state throughout the duration of the mission unless otherwise commanded to transition from the state. Such a commanded condition would be when the robot receives an indication to move to a new search region.

Both the parallel-transect and spiral patterns are implemented by stitching together individual points such that the desired pattern is formed. In either case, the robot computes a destination point based on the last point on the search pattern that it reached. The robot then drives to this destination point from the previous point on the path. The next point in the search pattern is computed upon reaching this intermediate point; Algorithm 3 provides a

pseudocode sketch of how these algorithms operate. The patroller pattern is relatively simple compared to the other two search patterns, *i.e.*, only two points ever exist for this search pattern. When the robot arrives at the point that is designated as its current destination, the search pattern cycles back to the previous start point. Overall, the algorithm remains the same as for the spiral and lawnmower search patterns. To navigate from point to point, the robot uses the same navigation algorithm that is used when it is in the *go-to-goal* state.

Algorithm 3 A sketch of the implementations of the search patterns.

Require: (x_r, y_r, ϕ_r) set to the current robot pose; (x_g, y_g) set to the current search pattern point; and D_{max} as a distance at which the pattern switches to the next point.

```
1:  procedure PatternControl(x_r, y_r, φ_r, x_g, y_g, D_max)
2:      d ← sqrt((x_g − x_r)^2 + (y_g − y_r)^2)
3:      if d < D_max then
4:          if Reverse pattern then
5:              Compute next "reverse" point (x_g, y_g).
6:          else
7:              Compute next "forward" point (x_g, y_g).
8:          end if
9:      end if
10:     Steer to goal point (x_g, y_g).
11: end procedure
```

For the spiral pattern, the next point is computed based on a synthetic time index, using the parametric equations for the arithmetic spiral, as given in Section 3.6.3. The index is an integer index that is incremented or decremented as the robot reaches destination points on the spiral. Once the time index has reached its maximum value, the robot travels along the reverse of the spiral path until it reaches its original starting point. The search process is then restarted after the robot reaches the starting point.

The parallel-transect pattern is slightly more complex than the spiral pattern. An additional state machine, illustrated in Figure 31, is used by the robot controller to determine whether or not a vertical or horizontal traversal is necessary. Depending on the current state of this state machine, the next point is either directly horizontal or vertical from the current robot position, or reversed along the current path of travel.

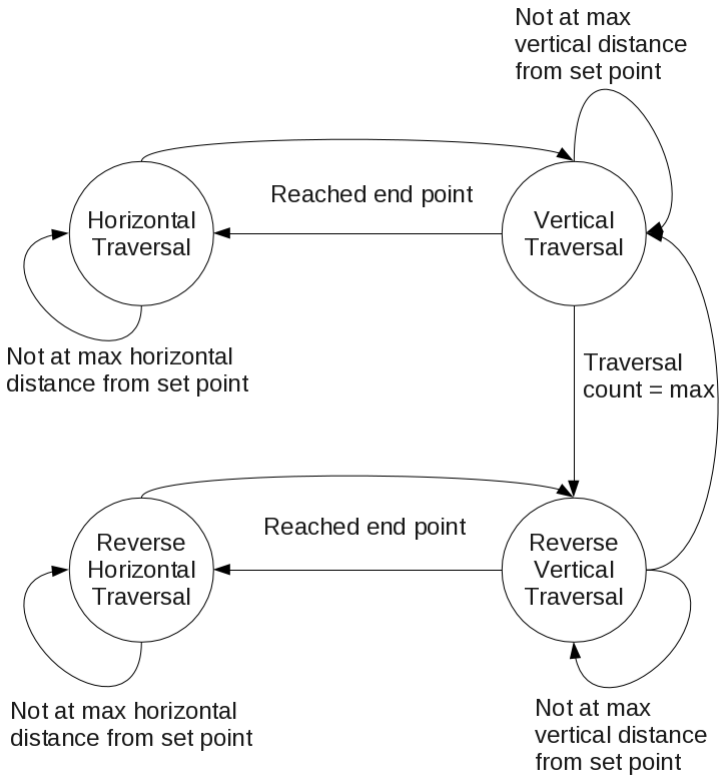


Figure 31. State machine for the parallel-transect search method.

The ablated-target sensor mounted to the robot monitors for any potential targets as the robot performs each sweep of the search area; if any targets are detected, the resulting target measurements are sent to the modeling component to add to the local and global ablation region models. In addition, the sensor error model may generate a false alarm rather than a true target measurement. Each of these false alarms are uniformly distributed within the field of view of the sensor, but as stated in the assumptions in Section 3.3.2, a false alarm will never be generated simultaneously with a target measurement.

The robot can collect multiple target contacts within its sensor field-of-view; the main component bundles all of these target contacts together, and posts the entire bundle of measurements to the list of measurements that is consumed by the modeling component.

5.1.4.4 Return-home State

The *return-home* state can only be entered upon receiving a command to do so, either by direct command issued by one of the other agents on the network, or from within the controller itself, if the controller determines that the robot cannot continue the mission (*e.g.*, the robot's energy has fallen below a certain threshold). This state allows the robot to return to a predefined home location, stored within its initial configuration and set at initialization time during the *start* state, and exit gracefully from the controller. Navigation to this home position follows the same navigation algorithm as in the *go-to-goal* state.

5.1.5 Configuration and Outputs

To define the behaviors and missions for the robot, the controller must accept several configuration files that detail this information. Three configuration files must be specified to the controller:

- *Robot limits*: contains the physical limits of the robot relevant to the controller, such as minimum and maximum turn rates and speeds. These limits assume unicycle-like or differential-drive-type robot platforms. That is, robots that use the type of control assumed in Section 5.1.1. While this is the model assumed, other types of robots can fit into this paradigm.

- *General controller configuration*: this file includes specialized parameters for the selected search method, as well as other controller parameters. For parallel-transect patterns, the file includes the default traversal distances. For spiral patterns, the file includes the spacing between turns and the spacing between points on the spiral. For a patroller, the direction and distance to be patrolled are the key parameters. General configuration options include the distance at which the robot should switch to new

waypoints, and whether or not the agent should travel to a separately defined goal point before initiating its search algorithm.

- *Mission file*: this file contains definitions of the agents in the mission that are used for communication between agents, and information on which agents are the arbiters and patrollers.

The robot controller also persists part of its internal state across missions for later analysis. This internal state consists of the parameters of the last mixture model that was generated by the agent, the iceberg position measurements used to generate that model, and the metrics that were computed from that model. To accommodate later analysis of the data, an Extensible Markup Language (XML) schema was developed to describe the data. The robot stores the data in this format upon termination of the controller. Figure 32 is an example output from the IIP dataset study in Section 3.5, if latitude and longitude were used as the measurements. As it shows, the resulting XML file contains the components of the mixture model, the extents of the search regions generated from it, the metrics, and the complete measurement history. This file is output for both the local and global model for each of the agents in the mission.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<gmodel valid="1">
  <startpos x="55.000000" y="-59.000000" />
  <components dim="2" lhood="-27936.222234" iters="149">
    <comp index="0">
      <weight>0.019058</weight>
      <avg_vel>0.010000</avg_vel>
      <avg_time>500</avg_time>
      <cost>0.000000</cost>
      <mean x="5.644478e+01" y="-5.909476e+01" />
      <covar>
        <entry x="1" y="1" value="3.257878e-02" />
        <entry x="1" y="2" value="0.000000e+00" />
        <entry x="2" y="1" value="0.000000e+00" />
        <entry x="2" y="2" value="1.569725e-03" />
      </covar>
    </comp>
<!-- Component list truncated -->
  </components>
  <history>
<!-- Measurement history has been truncated. -->
  </history>
  <regions>
    <region left="55.773895" top="-58.947492" right="57.115674"
      bottom="-59.242019" />
    <region left="55.014234" top="-59.290016" right="56.782876"
      bottom="-59.603262" />
    <region left="55.159702" top="-59.022707" right="55.781570"
      bottom="-59.516521" />
<!-- Search regions truncated -->
  </regions>
  <metrics>
    <count>230</count>
    <aicc>56243.902413</aicc>
    <meantime>0.000000</meantime>
    <totalmodelarea>7.358898</totalmodelarea>
    <unique>334</unique>
    <totaltargets>334</totaltargets>
    <avgtargetvel>0.034311</avgtargetvel>
    <coverage>0.000000</coverage>
    <predagent>20.000000</predagent>
    <agentcount>20.000000</agentcount>
  </metrics>
</gmodel>
```

Figure 32. Saved mixture model state XML file example.

5.2 Simulation Environment

For initial validation of the algorithms, a simulation environment was employed. The Player project and its associated robot simulator project, Stage, were used in developing the controller [84]. The Player project provided the libraries used for interfacing with the simulation and for interfacing with real robot hardware, which proved convenient when the controller was moved from simulation to hardware.

The robotic agents were modeled in Stage using a slightly modified version of the iRobot Roomba/Create model included with the Player/Stage source distribution. The Roomba/Create model is a differential-drive model, which adheres to the assumption on the robot's movement model (as discussed in Section 5.1.1). The agent's position is determined by calculation from odometry measured from simulated wheel encoders. The primary modification to the models was the addition of a fiducial detector. This fiducial detector acts as the sensor that each of the robots must have to detect targets; instead of detecting fiducials on larger obstacles or items in the environment, entire moving targets are designated as fiducials in and of themselves, which allows for the agents to identify and localize the targets within the sensor's field-of-view. In this case, as a result of the nature of the Stage fiducial detector, this field-of-view is a circle with a fixed radius centered at the robot.

To model the targets, an ablating source controller was developed using the Stage libraries. This controller can be attached to any object within the Stage environment, allowing any object to generate targets in a similar manner to a glacier that is calving icebergs. Each ablation point also generates targets of differing size and velocity to be comparable to the activity of realistic target sources at the glacier-sea interface. To provide for variations in ablating target sources and the targets that they generate, a set of parameters is associated with each ablation point. The target source can have many of these ablation points with varying behavior. These parameters include the following: positions of the ablation points on the source, the mass calving rate (in terms of tons of ice per target), the means

and standard deviations of the dimensions of new targets, the means and standard deviations of the velocities of the calved targets, and the correlation coefficients of the velocities with respect to other ablating sources. Adding correlation between target sources allows for noise in the velocity to be drawn from an l-dimensional multivariate Gaussian, where l is the number of ablation regions. Each target source will also have a maximum amount of mass that may be calved to form targets; once this mass has been ablated away, the target source can no longer generate new targets.

Setting these parameters and the positions of the target sources is accomplished by providing a configuration file to the controller. Different configuration files may be set for different target sources in the simulation, which allows for a multitude of different target sources to be placed within a single simulation. This feature may be used to test extremely hazardous environments; for example, cases where extremely large icebergs have been calved from a glacier, which themselves may more slowly generate icebergs as they break up over time.

A screenshot of an example simulation run is shown in Figure 33. The gray circles are the simulated agents, the blue blocks are the calved targets, and the red rectangle is the target source. There are four individual ablation regions on this target source, which accounts for the four target streams that may be seen. The dotted lines emanating from the two agents on the left side of the screenshot to targets that appear to be close to them graphically indicate the bearing and range of the agent to a target that falls within its sensor's field of view.

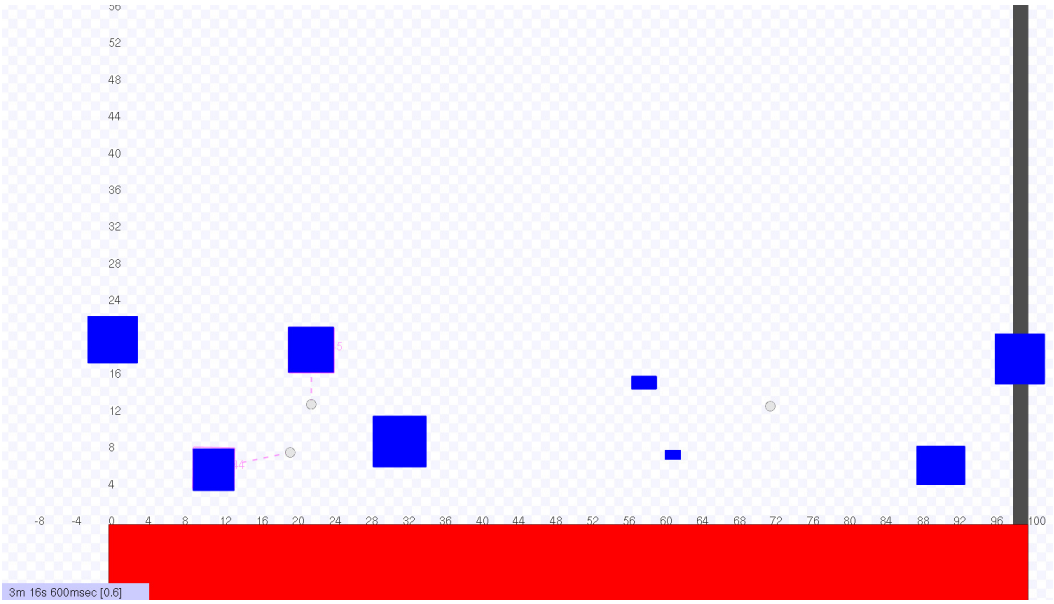


Figure 33. Screenshot of Stage simulation environment.

5.3 Scenario Replay Environment

For a more accurate and visually-pleasing presentation of the results of a simulation run, a scenario replay environment using the Unity3d engine was developed. Given a "unified" log file containing all of the agent and target positions from a given simulation run, this environment can reconstruct how the scenario developed over time. This environment does not execute a physics simulation of its own to replicate the scenario; it relies entirely on the contents of the log file to direct the objects contained in the scene. A screenshot of the environment is shown in Figure 34.

The agents are represented using generic UAV models, and the icebergs are represented by the variously colored iceberg models that float away from the larger glacial region. The colors of the icebergs depend on whether or not they have been detected by an agent: white indicates a target that has not been detected by any agent, magenta indicates a target that has been detected once, and blue indicates that it has been detected twice or more.

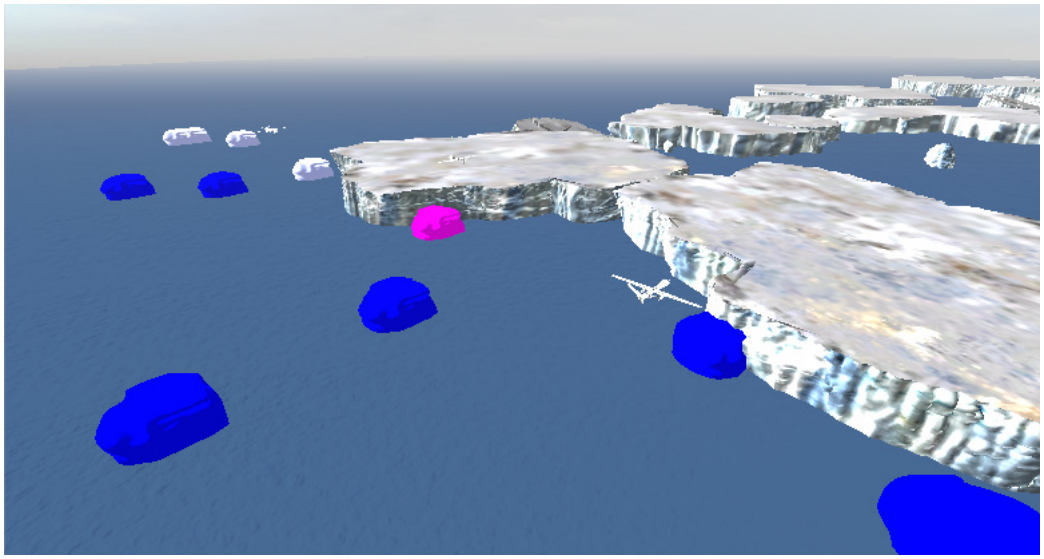


Figure 34. Screenshot of the scenario replay environment.

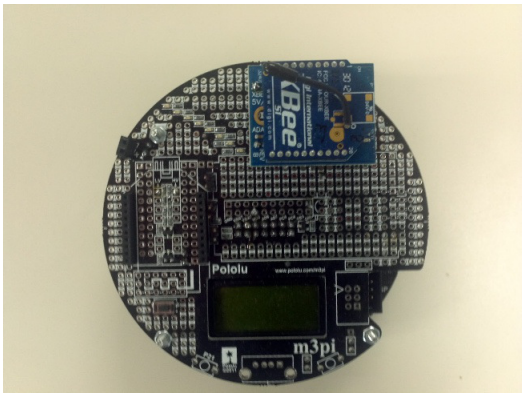
5.4 Hardware Platform and Software

To appropriately evaluate the algorithms in the real world, a hardware platform must be developed. While it would be appropriate to develop a platform that would operate in Arctic regions, a platform that can be used within a laboratory setting would also be desirable for initial testing. This was the goal in developing this particular multiagent platform. In this section, the design of the hardware used in the hardware platform is described, as is the design and implementation of the driver software used with the hardware.

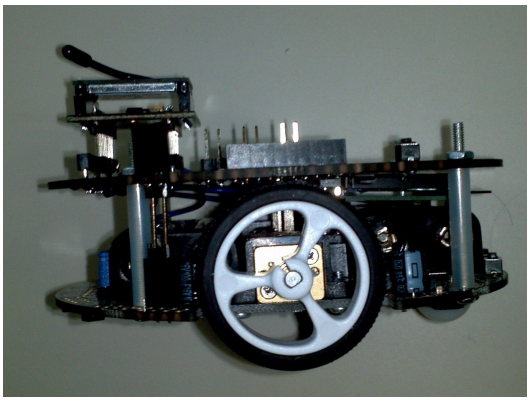
5.4.1 Robot Platform

The robotic base that was used is the Pololu 3pi robot. It is a small, two-wheeled, differential-drive, mobile platform that has several on-board resources driven by an Atmel ATmega328 (AVR) microcontroller. In addition, Pololu provides an expansion board to attach different types of hardware to the robot. In this case, the additional expansion board is used to attach an XBee radio transceiver for wireless communications between the robots. Since the robot base does not provide sufficient current on its own to drive the XBee transceiver, an adapter module produced by Parallax that provides sufficient current in addition to regulating the

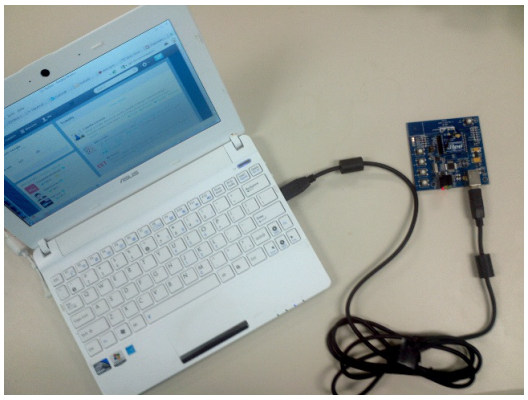
power supply voltage to 3.3V, as is required for the XBee, was installed between the Pololu 3pi and the XBee. The complete platform is shown in Figures 35(a) and 35(b).



(a) Modified Pololu 3pi for wireless communications.



(b) Pololu 3pi side view.



(c) Example base station for controlling the Pololu 3pi robots.

Figure 35. Pololu 3pi robot platform and base station.

Since the microcontroller used on the Pololu 3pi does not have sufficient resources to run the complete controller as previously described, the robots are driven by commands issued by a base station, which runs the controllers that issue the commands to the robots. There are no particular requirements for the base station, other than having the ability to run the necessary driver software and the robot controller; an example configuration is shown in Figure 35(c). The driver software and the firmware for the Pololu 3pi platform is

discussed in the next section.

5.4.2 Driver Software and Firmware

To use the Pololu 3pi robots with the robot controller, a software driver is necessary for communication between the base station and the robots. Firmware was written for the Pololu 3pi that allows for commanding the robot to perform different tasks and querying robot state that mimics the commands used by the K-Team Khepera robot. This command set uses a simple, string-based protocol. The benefit of using this command set is that the Player project has a driver for the Khepera, therefore using the Pololu 3pi robots with Player becomes straightforward. However, there are several issues that must be taken into account:

- The Player driver does not provide a way of obtaining the current energy for the robot through the standard Player interfaces. That is, a command to obtain the current robot battery power is not documented in the code.
- The physical dimensions of the Pololu 3pi are different from the Khepera. In addition, parameter scaling is performed within the driver that must be adjusted for the Pololu 3pi.
- The driver must be modified to allow for multiple robots of the same type to be driven using different commands from the same XBee device.
- Sensors to detect targets must be added to the robot, and an interface to the sensors must be added to the capabilities of the driver.

The first problem is easily resolved by adding the appropriate capabilities to the firmware. The second and third issues are resolved by modifying the Player driver and the associated configuration files such that the parameters are correct. In addition, the Khepera commands have been extended in a backward-compatible manner to add an optional robot identifier. If a robot receives a command that does not contain this identifier, then it executes the

command as normal. Otherwise, the robot checks to see if the identifier matches the identifier that has previously been stored within the microcontroller's electrically-eraseable, programmable, read-only memory (EEPROM). If the identifiers match, then the robot executes the command, otherwise the command is ignored.

The last issue, however, requires a different solution. There are many ways of simulating or implementing a target in a lab environment. Some examples include using other robots as targets and detecting them either using beacons or computer vision, making small motorized devices solely for the purpose of acting as targets, or devising a method of simulating the glacier-sea interface in a water tank. Since the types of robots that are being used here are ground-based, an augmented-reality solution to the target problem was used.

The augmented-reality target-simulator or virtual-target simulator runs as a server that broadcasts target positions to all of the robots. The target positions are broadcast using the same XBee device that transmits commands to the Pololu 3pi robots. The target positions are, in fact, similar in syntax to the Khepera commands, except with a command code that indicates that it is a target position.

To implement the capability of broadcasting these positions, socket communications are employed to deliver the packets to the robot-driver software. The target simulation is based on the same Stage controller that is used to implement an ablating-target source, which was described in Section 5.2. The same configuration-file format that is used to configure the Stage controller is used to configure the virtual-target simulator; this allows for scenarios to be developed using Stage and transferred directly to a test for a real hardware system.

The final issue is how to transfer the resulting measurements of target positions to the robot controller. The Khepera Player driver was modified to expose a fiducial marker device with behavior similar to that of the simulated device within Stage. The measured target positions are then scans returned to the controller through the fiducial interface obtained by the controller, and the scans are processed as measurements in the exact same manner as the simulated Stage scans.

CHAPTER 6

EXPERIMENTAL RESULTS

To provide an evaluation of the performance of the controller and the iceberg modeling and agent reallocation algorithms, a series of simulations were carried out using synthesized data that approximates the conditions and behavior of ablating target sources. In this chapter, the overall setup and results of these simulation studies are discussed. These studies include the following: the initial pilot-study that was conducted for the modeling algorithm; a simulation study examining the data coverage properties of the modeling methodology; a simulation study examining the target coverage properties of the modeling methodology; and a simulation study which contains the full implementation of ablating-source modeling and search-region assignment. Finally, tests were conducted using the real hardware platform described in Chapter 5.

6.1 Initial Pilot Simulation

The model generated by the GMM-based, multiple-agent, modeling algorithm is compared with that of a more traditional scanning method for icebergs using airborne radar in a simulation environment [85]. That is, a scan pattern composed of multiple parallel-transects across an area of interest, similar to the scan pattern used in the study described in reference [6]. This is a pilot simulation intended to determine the quality of models generated using the modeling techniques described in Section 3.4.

The necessary robot controllers and target sources were implemented and simulated using the software libraries provided by the Player/Stage project [84]. The metrics recorded as a result of the simulation to determine algorithm performance include target coverage and contact time. Target coverage is important to consider, because missed targets contribute to an incomplete model. Fifty trials were performed for each case, with the target source configured as shown in Figure 36, with target velocities that correspond to ocean

currents in the regions of interest. Each trial was time limited to approximately five minutes of collection time; the ablation rates of each activity point were thus accelerated to accommodate this time limit. The sizes of the targets that are calved from the activity regions were also varied; the source parameters are summarized in Table 7. Each parameter of the calved targets was modeled according to Gaussian errors, where (μ_{dims}, σ_{dims}) is the mean and standard deviation of the dimensions of each target, and (μ_{vel}, σ_{vel}) is the mean and standard deviation of the velocities. The mean sizes and velocities of the icebergs were chosen based on the frequency of classes of iceberg sizes and speeds [11]; smaller icebergs are more commonly encountered in practice. The ice mass from which the icebergs are calved was placed at the position $(x, y) = (50\text{ m}, -5\text{ m})$, with a length of 100 m, and a width of 10 m. Note that the width is the only "important" dimension for the ice mass; the length may be considered to measure the extent of the ice mass at the glacier-sea interface.

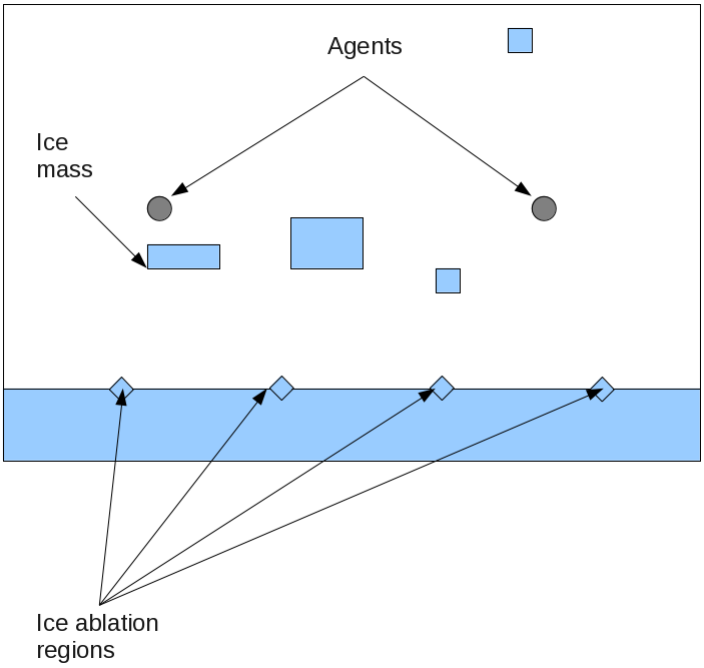


Figure 36. Target source configuration.

Each robot was equipped with a sensor that could detect a target within a circle that has a 10 m radius. For the multi-agent solution, each agent conducted its search using a parallel

Table 7. Summary of activity region parameters.

Active Region	μ_{dims}	σ_{dims}	μ_{vel}	σ_{vel}
1	5 m x 5 m	0.5 m	(-0.5, 0.5) m/s	0.05 m/s
2	5 m x 5 m	0.5 m	(-0.5, 0.5) m/s	0.05 m/s
3	2 m x 2 m	1.0 m	(0, 0.5) m/s	0.05 m/s
4	5 m x 5 m	0.5 m	(0.5, 0.5) m/s	0.05 m/s

transect pattern beginning at a starting point a distance away from the ablating source, within a rectangular cell. The agents began at points (x, y) = (18 m, 5 m), (40 m, 5 m), and (68 m, 5 m) respectively. In the four agent case, the agents were placed at (x, y) = (0 m, 5 m), (58 m, 5 m), (28 m, 5 m), and (85 m, 5 m). The single scanning agent was placed at the initial position (x, y) = (0 m, 5 m). All agents had a minimum speed of 0.1 m/s and a maximum speed of 1 m/s.

The basic metrics from the results of the simulations are given in Table 8. Average T_s is the average acquisition time of a target referenced from the time at which it is generated from a given activity point on an ablating source. Average model T_s is the mean acquisition time from the start of a given agent's mission; in this case, these times are referenced from the first agent. Percent coverage is defined as it is in Section 3.4.3: the ratio of the number of acquired targets to the number of total targets that have been generated. From these results and for this particular scenario, for near 100% coverage, three agents is sufficient, although the performance in terms of acquisition time suffers. Indeed, the average acquisition time is effectively doubled once a fourth agent is introduced, which lends additional empirical evidence that only three agents are sufficient. This doubling is the result of the further shrinking of the rectangular search regions in the given scenario: where one agent was reliably covering two of the ablating sources, further subdividing the regions only allows for partial coverage of one of the ablating sources by the additional agent. Hence, the lack of detected targets increases the overall acquisition time. Finally, given that the targets themselves are not fast-moving, the loss in acquisition time for the three agent case when compared to the two agent case, is not a significant issue.

Table 8. Simulation metrics summary.

Scenario	Pct. Coverage	Avg. T_s	Avg. Model T_s
Single Agent	21%	19 s	89 s
2 Agents	80%	6.5 s	58 s
3 Agents	97%	7.1 s	70 s
4 Agents	100%	14.1 s	77 s

In the case of the models generated, a resulting Gaussian mixture model that exhibits the following characteristics is desired:

- Sufficient target containment. That is, all targets, while perhaps not detected, can be accounted for within at least one component of the model at any point in time.
- Sufficient overall target coverage. Specifically, a model that results in maximized target coverage for a single agent. In addition, components that are well-separated in space will contribute to better identification of individual ablation regions and target streams, which will assist with target coverage.

Figures 37, 38, and 39 show a selection of the mixture models generated by each of the scenarios. Target measurements are denoted by stars; the target source is the gray rectangle, with activity points identified by the circles overlaid on the source. The Gaussian mixtures are represented by the heatmaps, with their means represented by the black crosses.

Compared with the other results, the mixture model generated by the single agent solution is not sufficient for this scenario: while there is good separation between components and given that the measurements are all clustered above a single source and it definitively identifies a particular activity region, there is insufficient target containment when all targets are taken into account. That is, Figure 37 only shows one of the four ablating sources; the targets generated by the other targets were not acquired by the single agent. In the multiagent cases, target containment is not an issue; search regions can be generated that can provide sufficient coverage for all of the activity points on the ice mass. Separation between components is also clear, providing for the generation of distinct search regions.

However, when generating search regions from the models, there may be a slight overlap between the regions assigned to particular agents. Additionally, for the model shown in Figure 39, one of the components may need to be inflated to provide proper coverage for a given region.

Overall, when taking the metrics into consideration with the resulting model, using multiple agents is the superior solution for model generation. This is true for target coverage, acquisition time, and model containment. However, depending on the scenario, it is important to consider the number of agents that are actually required. In the case here, while using three agents attained nearly 100% target coverage, the model that was generated using the two agent solution, as shown in Figure 39, provides sufficient target containment while sacrificing component separation and potentially resources, as a result of the size of the search regions that would be generated.

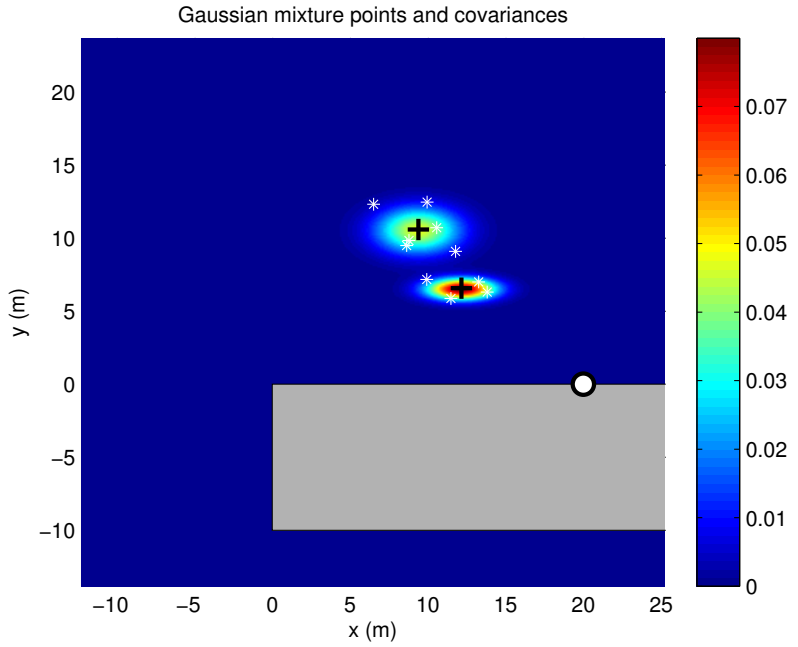


Figure 37. Gaussian mixture model resulting from a single-agent solution. Note only one target stream is represented.

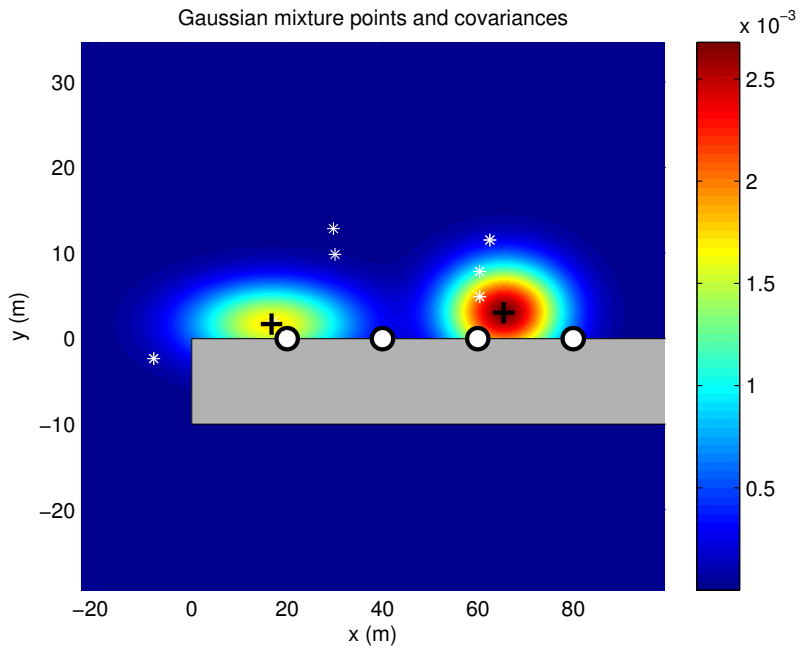


Figure 38. Gaussian mixture model resulting from a two-agent solution. There are two well-defined regions for reallocating sensors and capturing the behavior of the targets.

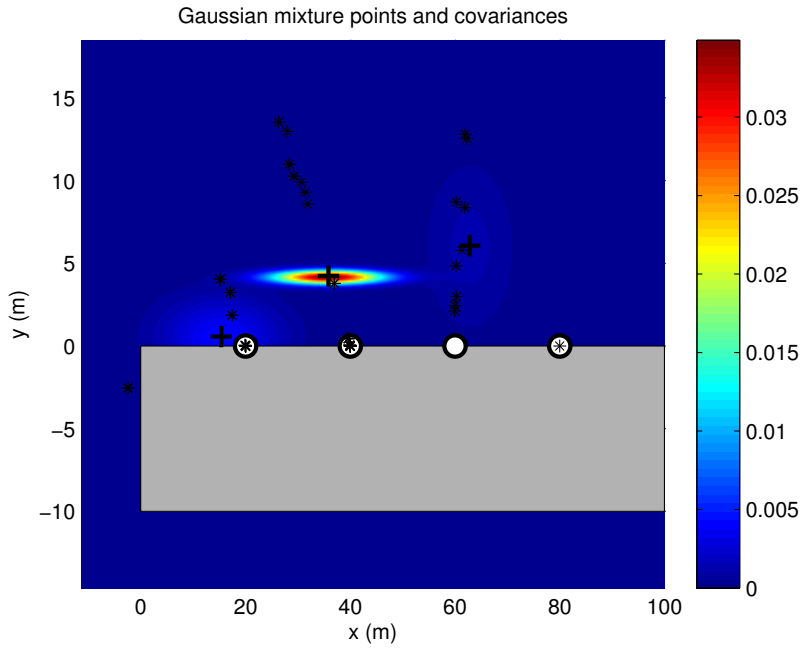


Figure 39. Gaussian mixture model resulting from a three-agent solution. All three regions amply capture the target behavior.

6.2 Data Coverage Evaluation

Based on the results of the pilot study, a means of determining the "goodness of fit" of a generated mixture model needs to be determined, in addition to determining how well a model fits future data from a given set of ablation regions. Hence, another simulation study was carried out to determine this goodness of fit.

To evaluate how well the modeling technique captures iceberg behavior in terms of data collected and data yet to be collected, a measure of similarity between the model and a given data set will be used. The model will first be compared with the target measurement data used to generate the model. Afterward, it will be compared with a subset of data obtained from target trajectories that may or may not have been used to obtain the target measurements; the trajectories will have similar target stream structure to that of the target streams used to generate the model.

6.2.1 Model Similarity Metrics

The following dimensionless similarity metric based on the Mahalanobis distance will be used for a single measurement:

$$\max D_s(Q, z) = \max_{j=1..k} \sqrt{\left(z - \mu_j\right)^T \Sigma_j^{-1} \left(z - \mu_j\right)}, \tag{48}$$

where z is a measurement in the data set, $Q = \left\{q_j\right\}_{j=1}^{k}$ is the model, k is the number of model mixture components, μ_j is the mean of component q_j, and Σ_j is the covariance matrix of q_j. In addition, the minimum Mahalanobis distance

$$\min D_s(Q, z) = \min_{j=1..k} \sqrt{\left(z - \mu_j\right)^T \Sigma_j^{-1} \left(z - \mu_j\right)}, \tag{49}$$

and the Mahalanobis distance using the *overall* mean and covariance of the mixture distribution

$$D_{s,ovr}(Q, z) = \sqrt{(z - \mu_{ovr})^T \Sigma_{ovr}^{-1} (z - \mu_{ovr})}, \tag{50}$$

where μ_{ovr} is the overall mean and Σ_{ovr} is the overall covariance of model Q, will be computed. The overall mean and overall covariance of a Gaussian mixture are computed as

follows:

$$\mu_{ovr} = \sum_{i=1}^{k} w_i \mu_i; \tag{51}$$

$$\Sigma_{ovr} = \sum_{i=1}^{k} w_i \Sigma_i + \sum_{i=1}^{k} w_i \left(\mu_i - \mu_{ovr}\right)\left(\mu_i - \mu_{ovr}\right)^T, \tag{52}$$

where the w_i are the component weights, the μ_i are the component means, and the Σ_i are the component covariance matrices. To compute the overall similarity between the model and the given data set, the average of the metrics plus a standard deviation will be computed. Verifying data coverage for a model will focus on 1) computing the similarity of a generated model to the given data set used to generate the model and 2) computing the similarity of a generated model to a different data set that contains measurements of targets newly ablated from the same sources used to generate the model.

Good model performance means that every measurement is covered by at least one model component q_j. The hypothesis is that the performance of the model, *i.e.*, how well it covers future data, is correlated with the spatial diversity of the target measurements. That is, if the original measurements are tightly clustered, then similarity between the model and new data will likely be reduced. Figure 40 illustrates this phenomenon. The gray dots are the points in an arbitrary data set, while the blue ellipses are the model components of a model generated from some previously acquired data set, specifically, the data set in the left-hand portion of the figure. The model is generated from highly clustered data; the center illustration indicates a dataset with lower similarity, while the rightmost illustration shows a data set with high similarity, with similar clustering as the original model's data set.

6.2.2 Test Scenarios

To evaluate the ablating source model with respect to data coverage, several permutations of the baseline scenario as given in Section 6.1 will be used. Note that "algorithms," in this case, refers to the methodology of obtaining measurements and updating the model. Each of these scenarios varies the target size and target velocity, two of the most important

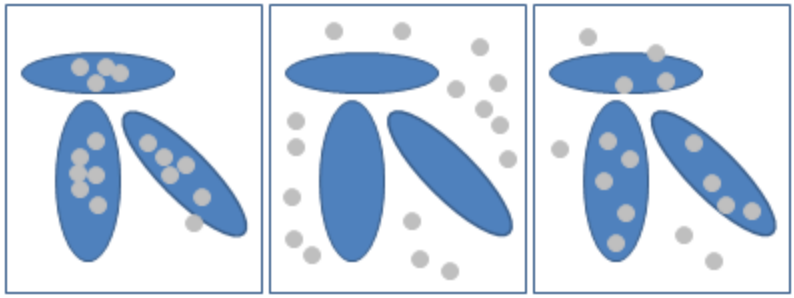


Figure 40. Illustration of different types of data sets as compared with a pre-generated model. The leftmost model is generated from tightly clustered data. The middle image illustrates the problems with a change in the ablation regions using a model generated with that type of data. The rightmost image illustrates more diverse measurements used in the model, which should improve its performance.

parameters when considering the detection of icebergs, as the smaller icebergs are the targets that are more difficult to detect, and unusual patterns in iceberg propagation may cause model irregularities.

The base scenarios are as follows:

- *Uniform target movement*: Target streams that move uniformly in a single direction at the same velocity, with no velocity variance.
- *Uniform target size*: All resulting targets are equal in size.
- *Target movement with uncorrelated velocity variance*: Each target stream has its own variance in velocity, with no correlation between individual target stream velocities.
- *Target movement with correlated velocity variance*: Ocean currents are not likely to have uncorrelated velocities in the x and y direction.

The uncorrelated and correlated velocity variance scenarios are collectively referred to as *non-uniform* scenarios.

The base scenario parameters are given in Tables 9, 10, and 11. Four ablation regions are used to correspond with the use of four agents. Ideally, when performing a reallocation using the data collected from the ablation regions, the assignment will result in each agent observing a different ablation region. The region of interest $\mathcal{S}$ is 100 m wide by 30 m high,

similar to the same region of interest for the pilot simulation in Section 6.1 as shown in Figure 36. The ablating sources are modeled according to Gaussian errors, where (μ_{dims}, σ_{dims}) is the mean and standard deviation of the dimensions of each target, and (μ_{vel}, σ_{vel}) is the mean and standard deviation of the velocities.

Agent starting (x, y) positions are given in Table 12 for mobile agents and Table 13 for fixed sensors. In each of these tables, "U. Size" refers to the uniform size scenario, "Lwm." refers to agents using the lawnmower search pattern, and "Ptl." refers to agents using the patroller search pattern. A mobile agent is defined as it is in Chapter 5. A fixed sensor is a mobile agent that has a constant angular velocity $\omega = 0$ and linear velocity $v = 0$. Note that in all scenarios, the fixed sensors are always in the same locations. Additionally, for the correlated velocity scenario, values for the velocity are drawn using the standard deviations in Table 11 according to a bivariate Gaussian distribution with nonzero correlation coefficients between *adjacent* ablating sources. Stacking the correlation coefficients used in this scenario into a matrix, indexed by the ablating source index i as defined in Section 3.3.1 by the definition of the ablating sources $U = \{u_i\}_{i=1}^{l}$, results in the following correlation matrices $\mathbf{R}_x$ and $\mathbf{R}_y$, one for each direction:

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0.1 & 0 & 0 \\ 0.1 & 1 & 0.3 & 0 \\ 0 & 0.3 & 1 & -0.3 \\ 0 & 0 & -0.3 & 1 \end{bmatrix}, \tag{53}$$

$$\mathbf{R}_y = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0.5 & 1 & 0.5 & 0 \\ 0 & 0.5 & 1 & 0.2 \\ 0 & 0 & 0.2 & 1 \end{bmatrix}. \tag{54}$$

Table 9. Summary of activity region parameters - Uniform Velocity.

Active Region	μ_{dims}	σ_{dims}	μ_{vel}	σ_{vel}
1	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0 m/s
2	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0 m/s
3	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0 m/s
4	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0 m/s

Table 10. Summary of activity region parameters - Uniform Size.

Active Region	μ_{dims}	σ_{dims}	μ_{vel}	σ_{vel}
1	5 m x 5 m	0 m	(0.2, 0.2) m/s	0.05 m/s
2	5 m x 5 m	0 m	(0.2, 0.2) m/s	0.05 m/s
3	5 m x 5 m	0 m	(0.2, 0.2) m/s	0.05 m/s
4	5 m x 5 m	0 m	(0.2, 0.2) m/s	0.05 m/s

Table 11. Summary of activity region parameters - Non-uniform Size and Velocity.

Active Region	μ_{dims}	σ_{dims}	μ_{vel}	σ_{vel}
1	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0.05 m/s
2	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0.05 m/s
3	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0.05 m/s
4	5 m x 5 m	0.5 m	(0.2, 0.2) m/s	0.05 m/s

Table 12. Agent starting positions - Non-fixed.

Agent	Uniform Lwm.	Uniform Ptl.	U. Size Lwm.	U. Size Ptl.	Non-uniform Lwm.	Non-uniform Ptl.
1	(0,5)	(0.25,15)	(0,5)	(0.25,15)	(0,5)	(0.25,15)
2	(58,5)	(23,15)	(58,5)	(23,15)	(58,5)	(23,15)
3	(28,5)	(46,15)	(28,5)	(46,15)	(28,5)	(46,15)
4	(85,5)	(68,15)	(85,5)	(68,15)	(85,5)	(68,15)

Table 13. Agent starting positions - Fixed.

Agent	Position
1	(13,21)
2	(38,21)
3	(61,21)
4	(87,21)

Each scenario is run for 30 trials for each scenario configuration, for 3000 simulation frames with a frame rate of 25 Hz. Scenario configurations differ based on sensor configurations: while the targets will behave according to the descriptions above, the sensors will be changed based on their configuration parameters. Additionally, the generated model for each trial will be compared with respect to the model similarity measures to *characteristic trajectories*, which is a set of target trajectories $B_i(t)$, for $i = 1...l$, that represents the future data from ablating sources that are identical in behavior to the original sources used to generate the models.

The sensor configurations will be changed for each of the scenarios as follows. The agents will use two of the scanning patterns in different scenarios: the lawnmower/parallel-transect pattern and the patroller pattern. Each of the agents' sensors will also have varying

sensor field-of-view radii for each set of trials for each scenario. The radii are as follows: 0.5 m, 1 m, 5 m, and 10 m. Each of the sensors will have a constant false alarm probability $P_{fa} = 10^{-3}$; typical constant P_{fa} values are approximately 10^{-6} [86], and hence this is a purposeful degradation in sensor performance to observe how false alarms affect model performance. Additionally, the sensors will have noise variances in each direction of $\sigma_x = 1$ m^2 and $\sigma_y = 1$ m^2. Note that the noise variance will cause detection errors greater than that of the field of view of one particular sensor configuration. This behavior is intended to model a sensor that has very poor detection capabilities.

Sensors that are stationary within a fixed-size cell are also used, to examine the benefits of using search patterns with "inadequate" sensors in comparison. These fixed sensors will have the same sensor properties as the mobile agents' sensors, with identical parameters varied over scenario configurations. The sensors are placed at the centers of the fixed-size cells.

The baseline or control scenario in all cases is a fixed-cell scanning approach for each of the sensors. That is, there is no use of the model to modify the sensor trajectories and allocation once the scanning mission has started. These fixed cells may be generalized as a centroidal Voronoi tessellation of the region of interest $\mathcal{S}$ near the glacier. The purpose of this comparison is to see if similar data coverage can be obtained for these "standard" scanning solutions, and could act as *a-priori* models for use with the modeling and reallocation algorithms.

6.2.3 Simulation Results

The results for each scenario configuration given varying search patterns and sensor fields of view are given in the following tables. Note that the greater the magnitude of this similarity metric, the more *dissimilar* the model is from the data, as the metric is a Mahalanobis distance. The first set of data is the similarity of the data set used to generate the models, while the second set of data shows the similarity of the characteristic trajectory for each

scenario to the generated models. Tables 14, 15, 16, and 17 provide the initial data set similarity, while Tables 24, 25, 26, and 27 show the similarity to the characteristic trajectories. Additionally, Tables 18 and 28 provides data coverage for two fixed sensor scenarios in the same manner: uniform target movement and uncorrelated target movement. In all of the tables, μ represents an average of the similarity, while σ is the standard deviation. Entries that show "N/A" are cases where there were not enough sensor measurements to compute a model, hence data similarity could not be computed.

To verify the statistical significance of the simulation results, a one-tailed t-test with null hypothesis mean 3.0, 29 degrees of freedom, and a significance level of $p < 0.01$ was conducted. The hypothesis mean was chosen based on the fact that the average Mahalanobis distance of a data set to a distribution that provides for a good fit will be within 3-σ. The t-scores and p-values for each scenario are shown in Tables 19, 20, 21, 22, and 23. With respect to the significance level, the results are statistically significant except for the majority of cases for min D_s where the sensor field-of-view radius is set to 1 m or 0.5 m. The results are also not statistically significant for the uniform scenario in control case using the lawnmower pattern with a 5 m field-of-view radius.

An examination of the raw data for the data set used to generate the models (Tables 14, 15, 16, and 17) shows that for the overall similarity $D_{s,ovr}$, all of the measurements are within, at most, 2-σ from the overall mixture distribution. This is a good fit, as one would expect for the original data set. The minimum similarity min D_s stays within the 3-σ range for the non-fixed sensor scenarios, except in most of the 0.5 m sensor radius cases, despite whether the reallocation algorithms are used or not. This result shows that although the target measurements may be within the area as defined by the overall distribution of the mixture model, the targets may not necessarily be strongly associated with any one component, especially in cases where the number of measurements acquired is very sparse, such as this small field-of-view radius case. Note also that this occurs almost exclusively when using the lawnmower search pattern; when using the patroller pattern, the models

Table 14. Model similarity - Control, lawnmower pattern, initial data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	1.47	0.939	95.9	180	1.40	0.669
Uniform	5 m	2.32	1.71	360	602	1.38	0.614
Uniform	1 m	1.90	1.18	63.8	50.1	1.34	0.514
Uniform	0.5 m	1.686	1.15	824	787	1.37	0.546
U. Size	10 m	1.40	0.649	30.5	17.7	1.27	0.663
U. Size	5 m	1.49	0.883	419	359	1.34	0.623
U. Size	1 m	1.93	1.49	1670	1350	1.38	0.647
U. Size	0.5 m	5.48	10.5	11700	12300	1.40	0.662
Uncorr.	10 m	1.40	0.639	29.6	15.0	1.26	0.661
Uncorr.	5 m	1.53	0.896	126	253	1.36	0.656
Uncorr.	1 m	1.82	1.26	1430	1070	1.37	0.647
Uncorr.	0.5 m	5.22	7.42	11600	6780	1.39	0.635
Corr.	10 m	1.45	0.733	28.9	15.4	1.34	0.636
Corr.	5 m	1.53	0.853	43.5	25.7	1.32	0.617
Corr.	1 m	1.71	1.08	900	945	1.38	0.520
Corr.	0.5 m	3.89	4.09	6710	4260	1.45	0.569

Table 15. Model similarity - Control, patroller pattern, initial data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	1.89	1.27	284	395	1.34	0.628
Uniform	5 m	1.69	1.08	392	466	1.35	0.611
Uniform	1 m	1.41	1.44	1200	1204	1.29	0.647
Uniform	0.5 m	1.51	1.01	2430	2280	1.29	0.677
U. Size	10 m	1.35	0.682	37.6	22.9	1.28	0.627
U. Size	5 m	1.37	0.768	44.9	39.4	1.43	0.679
U. Size	1 m	1.52	1.16	2410	2060	1.34	0.627
U. Size	0.5 m	2.32	2.46	10800	6690	1.43	0.643
Uncorr.	10 m	1.53	0.927	362	425	1.29	0.649
Uncorr.	5 m	1.48	0.827	528	644	1.30	0.655
Uncorr.	1 m	1.63	1.027	585	641	1.30	0.663
Uncorr.	0.5 m	1.61	0.985	425	699	1.33	0.668
Corr.	10 m	1.39	0.697	37.8	14.6	1.27	0.632
Corr.	5 m	1.43	0.807	209	347	1.43	0.694
Corr.	1 m	1.49	0.974	1814	1649	1.36	0.625
Corr.	0.5 m	1.94	1.76	6163	3533	1.43	0.653

Table 16. Model similarity - Using reallocation, lawnmower pattern, initial data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ min D_s	σ min D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	1.57	1.06	138	269	1.30	0.790
Uniform	5 m	1.91	1.40	537	611	1.31	0.762
Uniform	1 m	3.52	4.72	6043	6737	1.67	0.950
Uniform	0.5 m	3.81	6.81	11037	23204	1.45	0.675
U. Size	10 m	1.84	1.62	1706	1038	1.29	0.711
U. Size	5 m	2.00	1.59	1507	982	1.40	0.698
U. Size	1 m	1.94	1.43	2040	1938	1.43	0.688
U. Size	0.5 m	8.65	12.9	12577	9168	1.49	0.833
Uncorr.	10 m	1.70	1.46	936	645	1.28	0.698
Uncorr.	5 m	1.833	1.34	920	912	1.38	0.626
Uncorr.	1 m	2.30	1.96	2142	2028	1.43	0.665
Uncorr.	0.5 m	4.09	4.56	9642	6130	1.41	0.672
Corr.	10 m	1.43	0.794	61.6	80.4	1.34	0.687
Corr.	5 m	1.88	1.61	1000	1990	1.39	0.673
Corr.	1 m	1.74	1.10	952	959	1.39	0.591
Corr.	0.5 m	6.12	8.99	12700	6430	1.43	0.615

Table 17. Model similarity - Using reallocation, patroller pattern, initial data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	2.01	1.35	688	846	1.37	0.646
Uniform	5 m	1.70	1.09	266	388	1.36	0.661
Uniform	1 m	2.09	3.44	6100	4540	1.36	0.670
Uniform	0.5 m	2.73	5.78	5430	5680	1.35	0.747
U. Size	10 m	1.47	0.893	235	233	1.33	0.685
U. Size	5 m	1.45	0.887	175	307	1.38	0.698
U. Size	1 m	2.247	3.02	5740	5890	1.37	0.662
U. Size	0.5 m	2.73	2.73	6520	5010	1.42	0.712
Uncorr.	10 m	1.56	0.878	408	708	1.32	0.643
Uncorr.	5 m	1.48	0.812	383	342	1.30	0.635
Uncorr.	1 m	1.52	0.881	346	313	1.32	0.676
Uncorr.	0.5 m	1.51	0.816	215	456	1.30	0.636
Corr.	10 m	1.524	0.934	252	365	1.34	0.703
Corr.	5 m	1.47	0.885	96.34	181	1.38	0.675
Corr.	1 m	2.02	2.42	4002	4430	1.42	0.687
Corr.	0.5 m	2.65	3.01	6700	4720	1.46	0.704

Table 18. Model similarity - Fixed, initial data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	1.28	0.696	4400	3920	1.37	0.368
Uniform	5 m	1.48	1.83	6040	5640	1.25	0.707
Uniform	1 m	N/A	N/A	N/A	N/A	N/A	N/A
Uniform	0.5 m	N/A	N/A	N/A	N/A	N/A	N/A
Uncorr.	10 m	1.26	0.893	498	442	1.37	0.719
Uncorr.	5 m	1.40	0.931	3220	2404	1.22	0.701
Uncorr.	1 m	3.86	12.3	21800	46100	1.23	0.841
Uncorr.	0.5 m	9.60	41.8	18100	30400	4.13	34.7

Table 19. Model similarity p-values - Control, lawnmower pattern, initial data set.

Scenario	Sensor FOV	min D_s t-score	min D_s p-value	max D_s t-score	max D_s p-value	$D_{s,ovr}$ t-score	$D_{s,ovr}$ p-value
Uniform	10 m	-8.94	9.51e-10	5.39	1.20e-5	-13.1	1.02E-13
Uniform	5 m	-2.17	0.0413	5.45	1.01E-05	-14.4	7.71E-15
Uniform	1 m	-5.10	2.67E-05	5.15	2.33E-05	-17.7	3.20E-17
Uniform	0.5 m	-6.28	1.01E-06	5.46	9.90E-06	-16.3	2.93E-16
U. Size	10 m	-13.5	4.25E-14	4.55	0.000123	-14.3	9.99E-15
U. Size	5 m	-9.39	3.17E-10	5.43	1.06E-05	-14.6	5.92E-15
U. Size	1 m	-3.92	0.000680	5.46	9.66E-6	-13.7	2.97E-14
U. Size	0.5 m	1.29	0.171	5.47	9.37E-6	-13.3	6.59E-14
Uncorr.	10 m	-13.7	2.87E-14	4.38	0.000192	-14.4	9.01E-15
Uncorr.	5 m	-8.97	8.86E-10	5.41	1.12E-5	-13.6	3.36E-14
Uncorr.	1 m	-5.15	2.34E-5	5.46	9.75E-6	-13.8	2.40E-14
Uncorr.	0.5 m	1.64	0.104	5.47	9.40E-6	-13.9	2.17E-14
Corr.	10 m	-11.6	2.07E-12	4.41	0.000178	-14.3	9.99E-15
Corr.	5 m	-9.46	2.66E-10	4.84	5.53E-5	-14.9	3.34E-15
Corr.	1 m	-6.52	5.15E-7	5.46	9.80E-6	-17.1	8.61E-17
Corr.	0.5 m	1.19	0.192	5.47	9.44E-6	-14.9	3.42E-15

Table 20. Model similarity p-values - Control, patroller pattern, initial data set.

Scenario	Sensor FOV	min D_s t-score	min D_s p-value	max D_s t-score	max D_s p-value	$D_{s,ovr}$ t-score	$D_{s,ovr}$ p-value
Uniform	10 m	-4.77	6.63E-05	5.43	1.04E-05	-14.4	7.12E-15
Uniform	5 m	-6.63	3.77E-07	5.44	1.03E-05	-14.7	4.19E-15
Uniform	1 m	-6.06	1.83E-06	5.46	9.70E-06	4.4	7.92E-15
Uniform	0.5 m	-8.11	7.47E-09	5.47	9.53E-06	-13.8	2.41E-14
U. Size	10 m	-13.2	7.26E-14	4.75	6.87E-05	-15.0	2.83E-15
U. Size	5 m	-11.6	2.09E-12	5.06	2.98E-05	-12.6	2.35E-13
U. Size	1 m	-6.98	1.49E-07	5.46	9.55E-06	-14.4	7.56E-15
U. Size	0.5 m	-1.52	0.124	5.47	9.40E-06	-13.4	5.53E-14
Uncorr.	10 m	-8.68	1.76E-09	5.43	1.04E-05	-14.4	7.95E-15
Uncorr.	5 m	-10.0	6.10E-11	5.45	1.00E-05	-14.1	1.22E-14
Uncorr.	1 m	-7.32	5.87E-08	5.45	1.00E-05	-14.0	1.66E-14
Uncorr.	0.5 m	-7.75	1.90E-08	5.45	9.97E-06	-13.6	3.14E-14
Corr.	10 m	-12.6	2.45E-13	4.35	0.000210	-14.9	2.97E-15
Corr.	5 m	-10.6	1.71E-11	5.42	1.06E-05	-12.3	4.11E-13
Corr.	1 m	-8.48	2.95E-09	5.46	9.60E-06	-14.3	9.04E-15
Corr.	0.5 m	-3.30	0.00324	5.47	9.46E-06	-13.1	8.35E-14

Table 21. Model similarity p-values - Using reallocation, lawnmower pattern, initial data set.

Scenario	Sensor FOV	min D_s t-score	min D_s p-value	max D_s t-score	max D_s p-value	$D_{s,ovr}$ t-score	$D_{s,ovr}$ p-value
Uniform	10 m	-7.44	4.29E-08	5.41	1.10E-05	-11.7	1.51E-12
Uniform	5 m	-4.23	0.000286	5.45	1.00E-05	-12.1	6.81E-13
Uniform	1 m	0.600	0.328	5.47	9.40E-06	-7.66	2.40E-08
Uniform	0.5 m	0.655	0.317	5.47	9.35E-06	-12.5	3.04E-13
U. Size	10 m	-3.92	0.000668	5.46	9.76E-06	-13.1	8.22E-14
U. Size	5 m	-3.46	0.00219	5.04	9.78E-06	-12.5	3.15E-13
U. Size	1 m	-4.04	0.000479	5.46	9.56E-06	-12.5	3.25E-13
U. Size	0.5 m	2.399	0.0261	5.47	9.38E-06	-9.89	9.55E-11
Uncorr.	10 m	-4.85	5.21E-05	5.45	1.00E-05	-13.4	4.98E-14
Uncorr.	5 m	-4.76	6.81E-05	5.45	9.82E-06	-14.1	1.25E-14
Uncorr.	1 m	-1.95	0.0619	5.46	9.55E-06	-12.9	1.41E-13
Uncorr.	0.5 m	1.302	0.168	5.47	9.41E-06	-12.9	1.36E-13
Corr.	10 m	-10.8	1.20E-11	5.27	1.65E-05	-13.2	7.53E-14
Corr.	5 m	-3.80	0.000911	5.46	9.55E-06	-13.1	9.29E-14
Corr.	1 m	-6.29	9.73E-07	5.46	9.79E-06	-14.9	3.25E-15
Corr.	0.5 m	1.901	0.0678	5.47	9.40E-06	-13.9	1.92E-14

Table 22. Model similarity p-values - Using reallocation, patroller pattern, initial data set.

Scenario	Sensor FOV	min D_s t-score	min D_s p-value	max D_s t-score	max D_s p-value	$D_{s,ovr}$ t-score	$D_{s,ovr}$ p-value
Uniform	10 m	-4.02	0.000507	5.45	9.85E-06	-13.8	2.45E-14
Uniform	5 m	-6.48	5.72E-07	5.43	1.05E-05	-13.5	3.85E-14
Uniform	1 m	-1.45	0.138	5.47	9.43E-06	-13.4	5.22E-14
Uniform	0.5 m	-0.254	0.382	5.47	9.41E-06	-12.1	7.04E-13
U. Size	10 m	-9.41	3.01E-10	5.40	1.13E-05	-13.3	6.11E-14
U. Size	5 m	-9.54	2.20E-10	5.42	1.08E-05	-12.7	2.07E-13
U. Size	1 m	-1.36	0.154	5.47	9.41E-06	-13.4	4.80E-14
U. Size	0.5 m	-0.53	0.341	5.47	9.42E-06	-12.1	6.85E-13
Uncorr.	10 m	-8.96	8.95E-10	5.45	9.96E-06	-14.3	9.11E-15
Uncorr.	5 m	-10.2	3.81E-11	5.42	1.06E-05	-14.6	5.57E-15
Uncorr.	1 m	-9.19	5.06E-10	5.42	1.08E-05	-13.5	3.91E-14
Uncorr.	0.5 m	-10.0	7.19E-11	5.44	1.03E-05	-14.6	5.69E-15
Corr.	10 m	-8.65	1.91E-09	5.43	1.05E-05	-12.9	1.42E-13
Corr.	5 m	-9.44	2.75E-10	5.38	1.20E-05	-13.1	9.20E-14
Corr.	1 m	-2.20	0.038	5.47	9.43E-06	-12.6	2.63E-13
Corr.	0.5 m	-0.64	0.320	5.47	9.43E-06	-12.0	9.11E-13

Table 23. Model similarity p-values - Fixed, initial data set.

Scenario	Sensor FOV	min D_s t-score	min D_s p-value	max D_s t-score	max D_s p-value	$D_{s,ovr}$ t-score	$D_{s,ovr}$ p-value
Uniform	10 m	-13.5	4.55E-14	5.47	9.45E-06	-24.3	4.33E-21
Uniform	5 m	-4.54	0.000125	5.47	9.42E-06	-13.5	4.47E-14
Uniform	1 m	N/A	N/A	N/A	N/A	N/A	N/A
Uniform	0.5 m	N/A	N/A	N/A	N/A	N/A	N/A
Uncorr.	10 m	-10.7	1.56E-11	5.44	1.04E-05	-12.4	3.95E-13
Uncorr.	5 m	-9.42	2.95E-10	5.47	9.52E-06	-13.9	2.16E-14
Uncorr.	1 m	0.380	0.367	5.48	9.35E-06	-11.5	2.47E-12
Uncorr.	0.5 m	0.866	0.270	5.48	9.35E-06	0.178	0.389

produced constrain the minimum similarity min D_s within the 3-σ range. This is likely the result of the fact that the patrollers typically produce models with small, highly-separated components. Figures 41 and 42 demonstrate this phenomenon. The white dots indicate target measurements, while the heatmaps indicate the likelihood maps of the mixture models. Figure 41 is a plot of a mixture model generated from a uniform velocity scenario using the lawnmower pattern with a 10 m sensor field-of-view radius. There are two "weak" components near the center, while most of the probability is contained in smaller components on the left- and right-hand sides of the plot. On the other hand, Figure 42 is a mixture model generated from a patroller run with a 10 m sensor field-of-view radius. As can be seen, there are small components sharing much of the mixture probability more equally than for the lawnmower case. Regardless, the measurements are very sparse since the 0.5 m sensor does not detect very many targets, but the patroller results fit the target streams much more closely than the lawnmower pattern does.

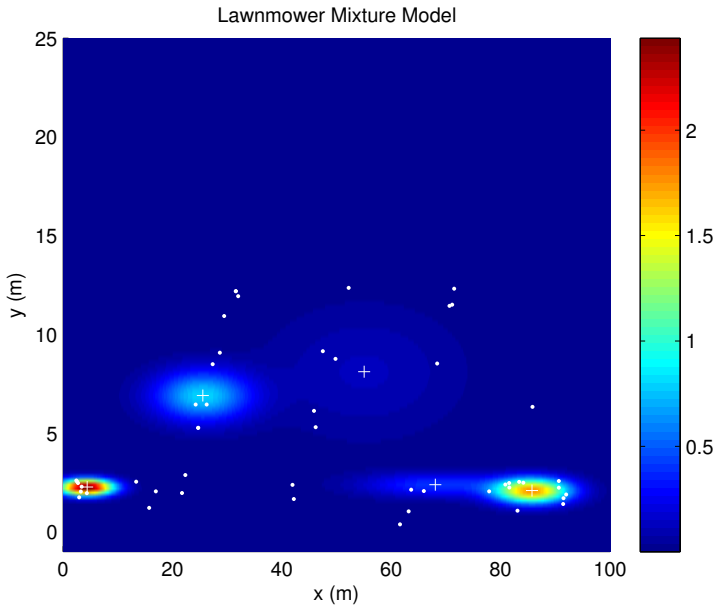


Figure 41. Lawnmower mixture model for uniform velocity scenario. Note that the components are spread out across the target streams.

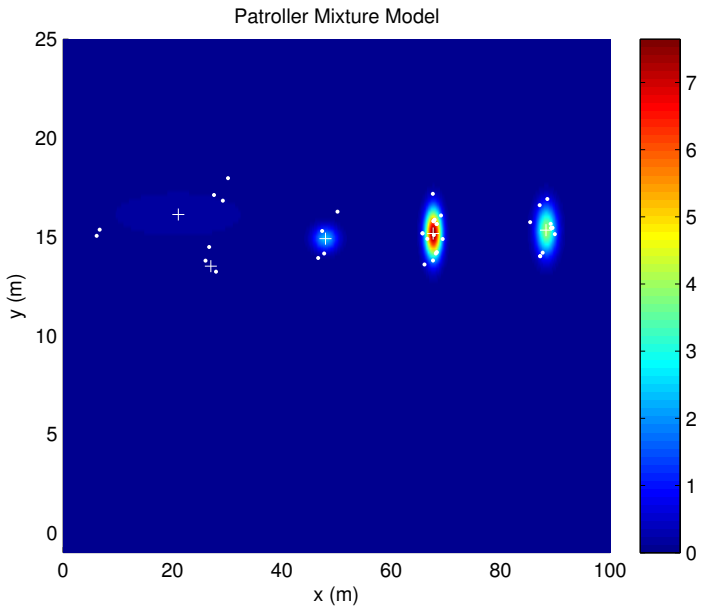


Figure 42. Patroller mixture model for uniform velocity scenario. Note that components are centered directly on each target stream.

One other case where this occurs is in the uncorrelated fixed sensor case, shown in Table 18. The distance $\min D_s$ is much greater than 3-σ, indicative of the sparsity of the measurements and the inability of the fixed sensors to acquire targets. A more curious problem occurs for the uniform case, as the targets are drifting toward the sensors. Although the fixed sensors are able to acquire some targets in this scenario, as will be seen in Section 6.3, they are not sufficient to develop a mixture model, as indicated by the "N/A" entries in the table. This is a consequence of the nature of the scenario and the placement of the sensors: quite obviously, if the sensors are not placed such that they can acquire the targets, then they cannot acquire them. This result shows the clear benefits of scanning with these types of sensors. These issues could be mitigated by increasing the sensor range (expensive) or by simply relying on existing satellites, with a time delay if the satellite is not geostationary with respect to the region of interest.

While the minimum and overall similarities $\min D_s$ and $D_{s,ovr}$ indicate how well the model fits the data, the maximum similarity $\max D_s$ indicates the *failure* of the model to

fit the data, which is just as important. In fact, the maximum similarity fully reveals the inability of the sensors with smaller field-of-view radii to produce models that fully reflect the state of the targets. For example, for the patrollers, it can be seen that the minimum similarity $\min D_s$ stays within the acceptable limits for both the control and reallocation algorithms. However, if one examines the maximum similarity over the scenarios and field-of-view radii, the rate of growth is extremely fast with a high standard deviation as the field-of-view radius shrinks. This behavior is better illustrated by plotting the similarity data to view these trends, as shown in Figures 43 and 44 for the minimum similarity means and Figures 45 and 46 for the maximum similarity means.

For the minimum similarity means (Figures 43 and 44), the plots remain relatively flat until the 1 m radius sensor is used. The patroller scenarios do not grow as rapidly as the lawnmower scenarios. For the maximum distance means (Figures 45 and 46), the growth is not as consistent for the scenarios that use the algorithms. Indeed, the profile of growth for both the minimum and maximum similarity for the control scenarios is very similar. For the maximum similarity for the scenarios using the reallocation algorithms, the inflection point is at 1 m for most of the scenarios, except for the uniform target velocity scenarios. The inflection point for these scenarios is at the 5 m radius; this is the result of early reallocation moving the agents away from one or more target streams. The other scenarios have a target spread as a result of the non-zero variances in the target velocities, which allow for mixture model components (and hence, search regions) to cover target measurements from ablating sources that are far apart. This cannot occur for the uniform velocity scenario.

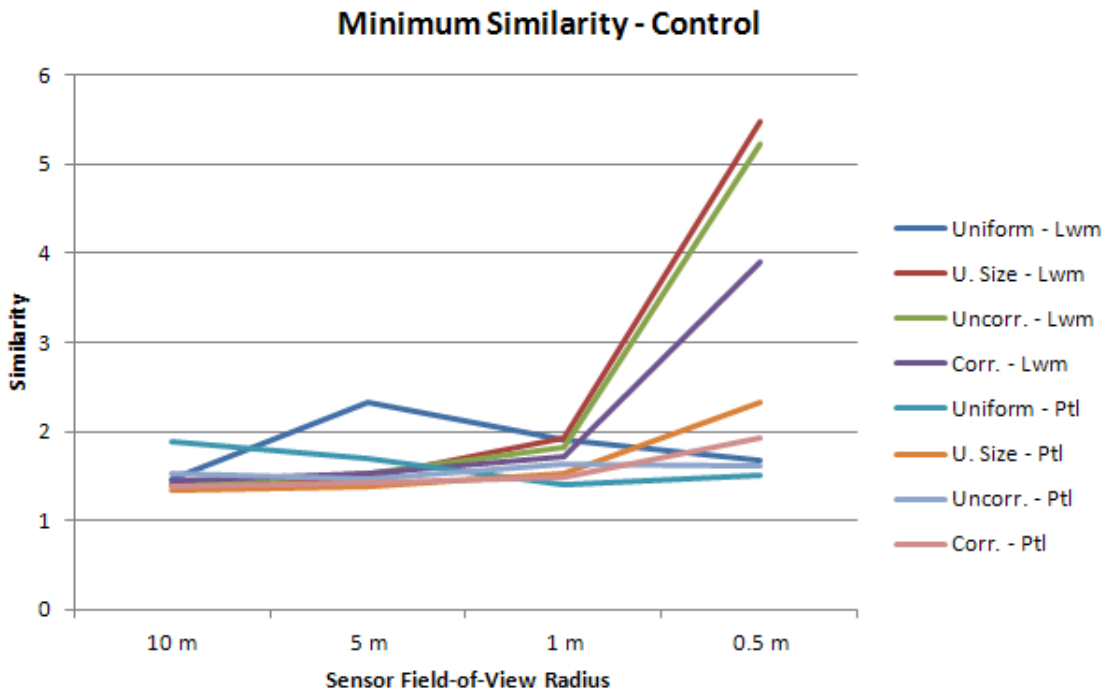


Figure 43. Plot of minimum similarity measures for the control scenario. This plot shows the apparent dependence of the similarity on the field-of-view radius, and only some dependence on search pattern.

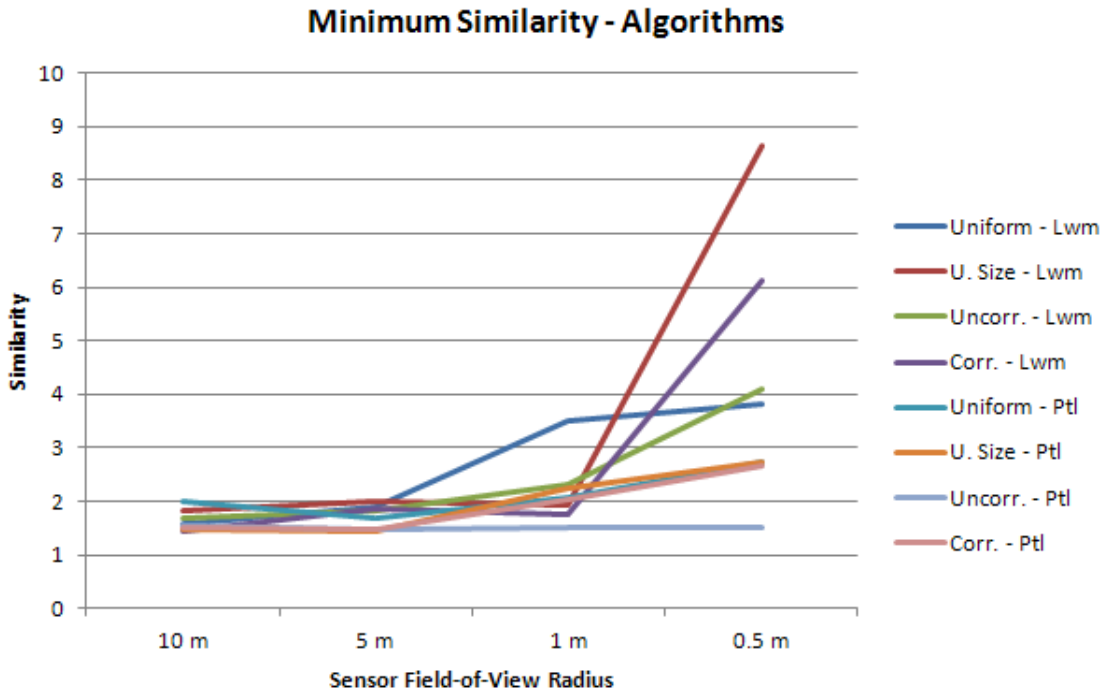


Figure 44. Plot of minimum similarity measures for the reallocation algorithms. For the reallocation algorithms the apparent dependence on sensor field-of-view is weaker, and more of a dependence on search pattern exists.

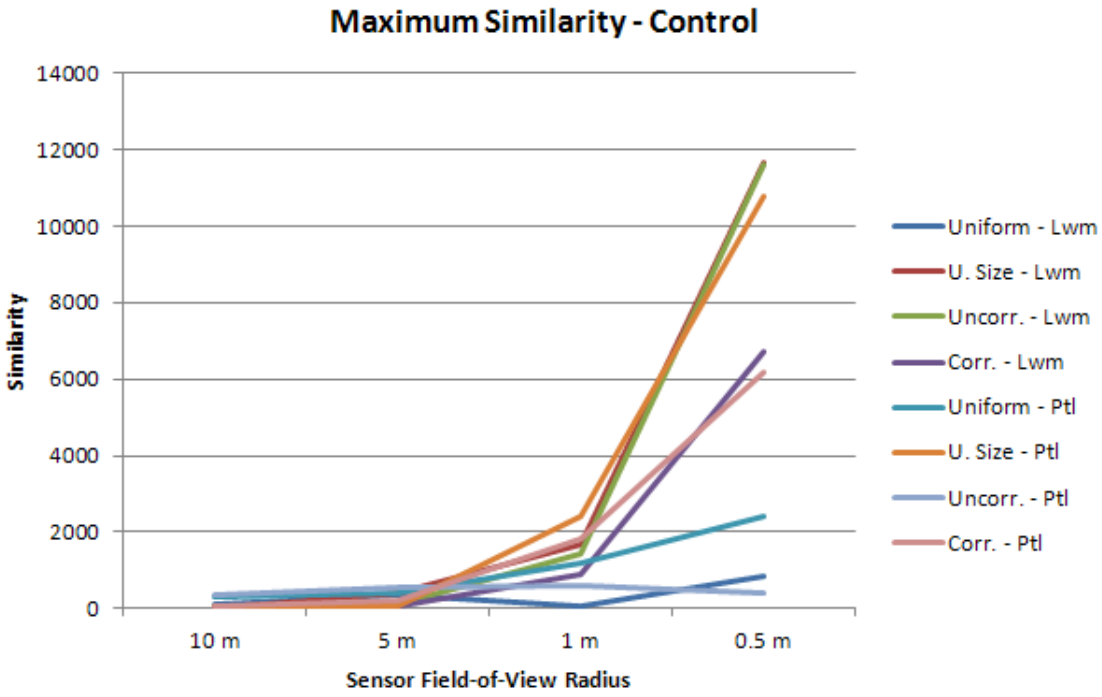


Figure 45. Plot of maximum similarity measures for the control scenario. An apparent dependence on the sensor field-of-view radius is shown, but the overall behavior depends on the scenario more strongly than the search pattern.

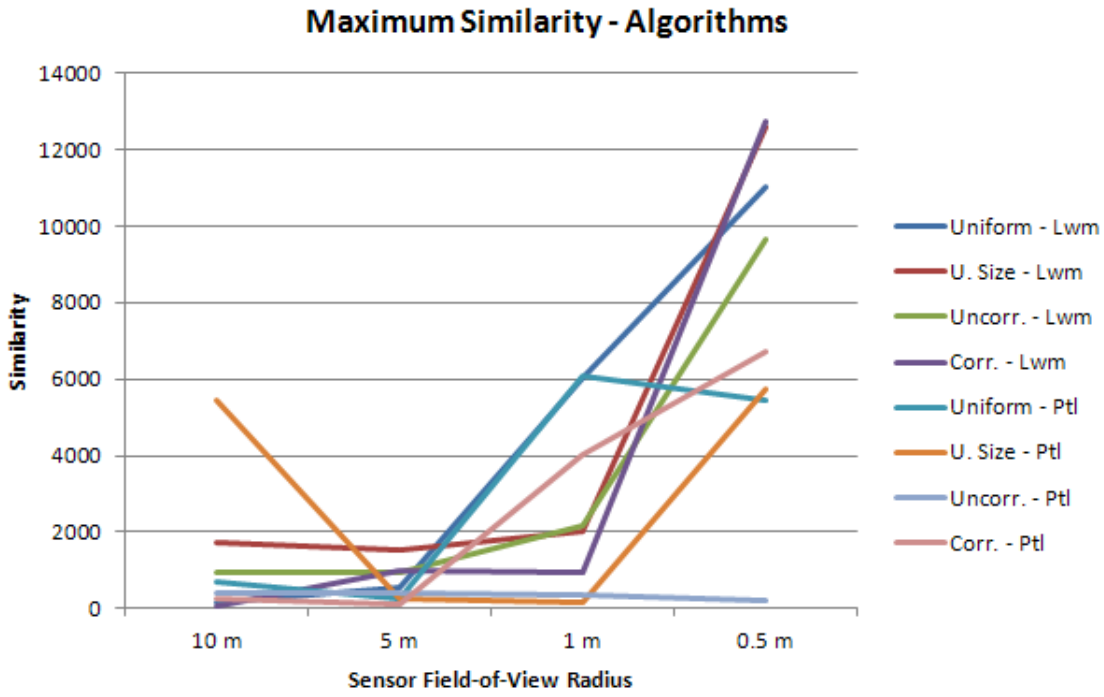


Figure 46. Plot of maximum similarity measures for the reallocation algorithms. The results are less consistent overall than the control, but there is still an apparent dependence on the sensor field-of-view radius, but less of a dependence on the search pattern and scenario.

Next, the models generated must be compared with the characteristic trajectories and determine the similarity of this data set to the models. The characteristic trajectories are shown in Figures 47, 48, 49, and 50. Note the overall similarity between the last three: the uniform size scenario uses the same velocity variances as the uncorrelated velocity, except the targets are all the same size. The correlated velocities for the trajectories in Figure 50 cause the trajectories to bend, rather than remain straight, as with the other two trajectories that have variance in their velocities.

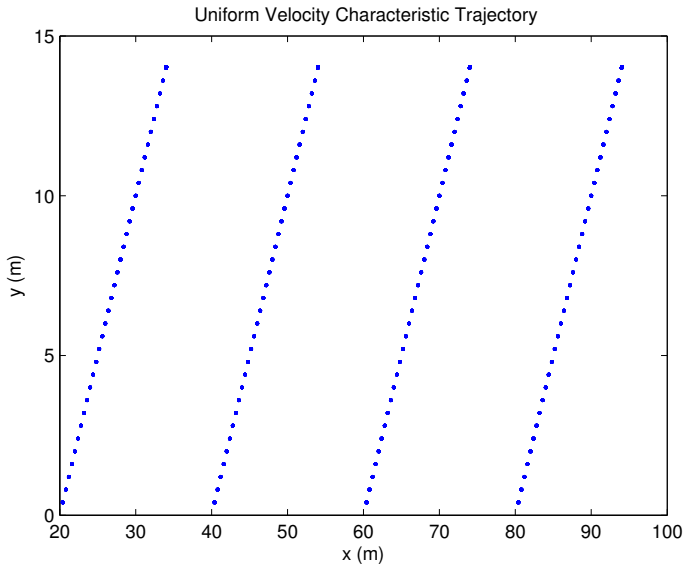


Figure 47. Uniform velocity characteristic trajectory. Note the sparsity of the trajectories.

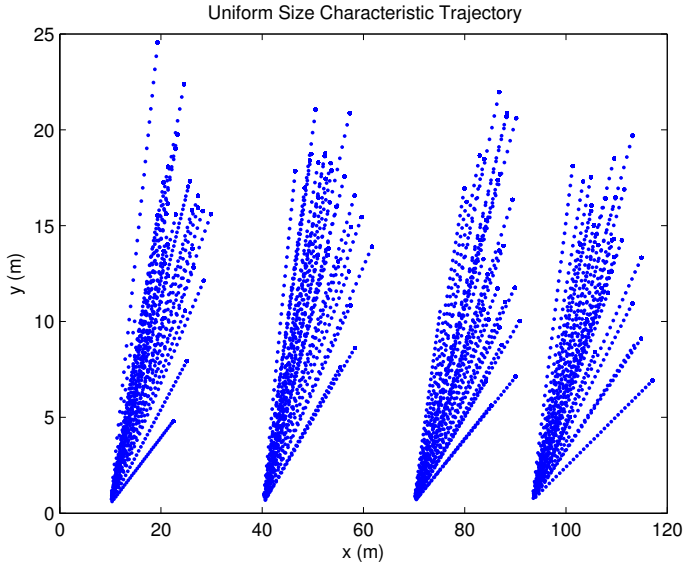


Figure 48. Uniform size characteristic trajectory. Note the similarity to the uncorrelated velocity trajectory; it is not shown, but each of the resulting targets are the same in terms of dimensions.

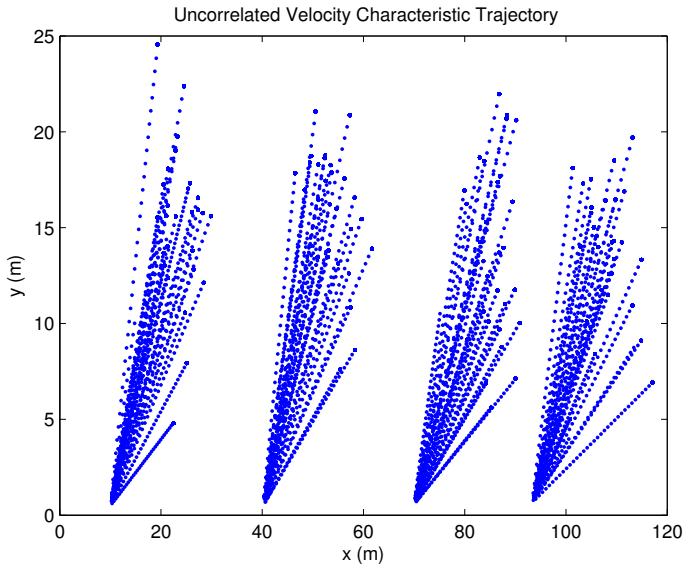


Figure 49. Uncorrelated velocity characteristic trajectory. The result is similar to that of the uniform size trajectory.

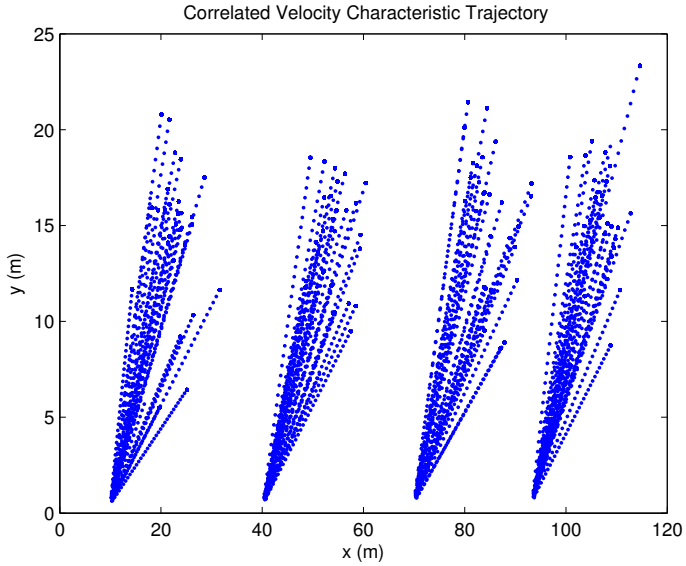


Figure 50. Correlated velocity characteristic trajectory. Note the bending in the trajectory paths.

For each of these trajectories, the corresponding similarities were computed with respect to the scenario that generated their respective models. The results are given in Tables

24, 25, 26, and 27. The first results to examine are the overall similarities $D_{s,ovr}$. As before, with the data sets that generated the models, the overall similarity is within 3-σ of the model. Figures 51 and 52 show the trend of the overall similarity to stay within 1-σ and 2.5-σ for the characteristic trajectories, which is reasonable.

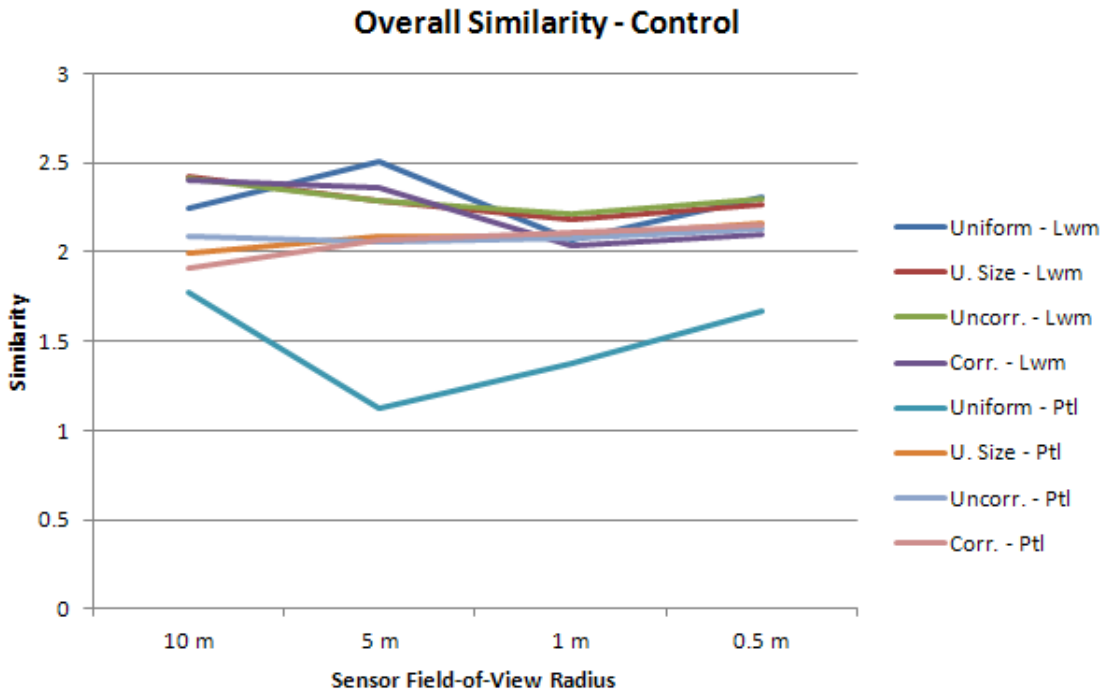


Figure 51. Overall similarity to the characteristic trajectories for the control scenarios. The results stay within the range 1-σ and 2.5-σ, but the models generated for the uniform case using the patroller pattern fit very well.

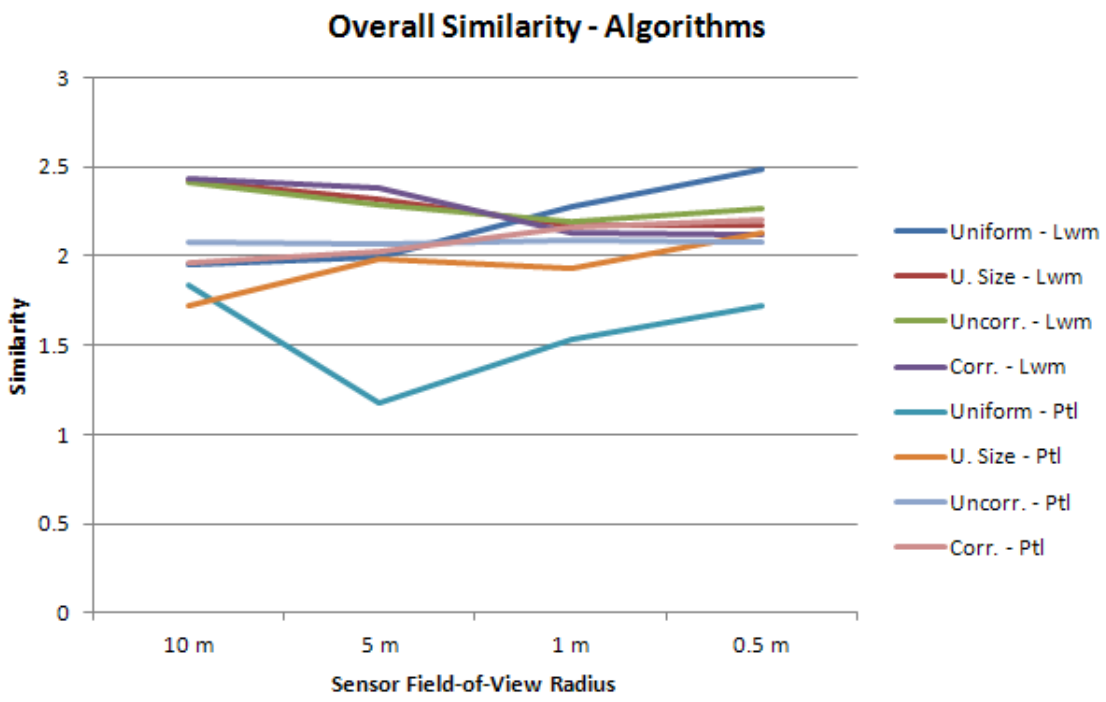


Figure 52. Overall similarity to the characteristic trajectories for the reallocation algorithm scenarios. Like the control scenario, the results stay within the range 1-σ and 2.5-σ, but the models generated for the uniform case using the patroller pattern fit very well.

Examining the trends for the minimum similarity $\min D_s$ and maximum similarity $\max D_s$ shows similar trends as those for the data sets used to generate the models, with some differences in the resulting magnitudes of the similarity. Figures 53, 54, 55, and 56 illustrate the data. The break points are at similar locations; however, the overall magnitude of the similarity is higher than the similarity for the original data sets, especially for the minimum similarity. The minimum similarity tends to range between 2-σ and 4-σ until the 1 m radius break point. While not a good fit on the high side of the range, as before, the trends remain the same. Examining the maximum similarity $D_{s,max}$, it is clear that the failure of the characteristic trajectories to fit the data is similar, if not identical, to that of the original data sets.

Finally, the fixed sensor model similarity must be considered. As before, for the uniform velocity scenario, the models fail to fit anything beyond the 10 m sensor field-of-view.

In addition, the uncorrelated velocity models do not fit very well, either: despite the overall similarity being less than 3-σ, the minimum similarity ranges from 3.55-σ to 25.3-σ. This suggests that models generated using fixed sensors are satisfactory for characterizing the target activity at the moment they are generated, but are not good for overall target modeling. This can be corrected: if the models are used as *a-priori* distributions for the models used by multiple agents, more spatially diverse measurements can be obtained and improve the data coverage of the models. This underlines one problem with using fixed sensors: the spatial diversity of the measurements is limited. The models generated by the nonfixed agents provided better overall data coverage, and models generated using either the "standard" scanning solution of fixed cells and the modeling and reallocation algorithms provided near equivalent data coverage for sensor fields-of-view that would be reasonably used in a scenario. In nearly all cases, both cases failed similarly when using substandard sensors.

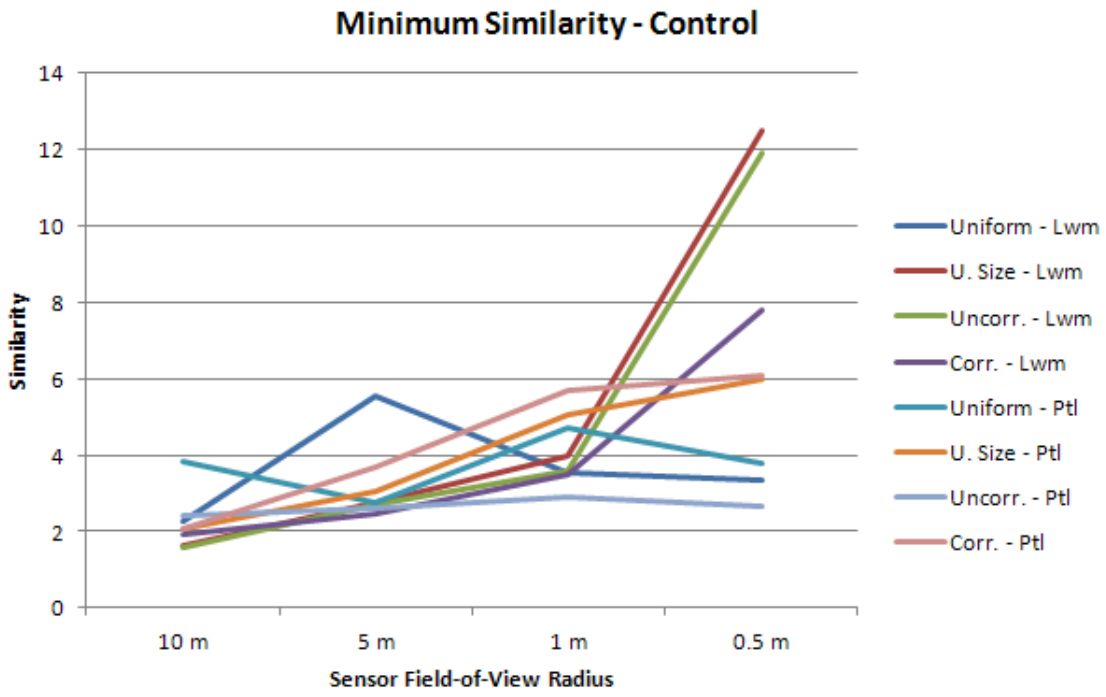


Figure 53. Minimum similarity measures for the control scenario and characteristic trajectories. Note that the overall similarity is worse (greater) compared to the data set used to generate it, in Figure 43, but the plot structure is similar.

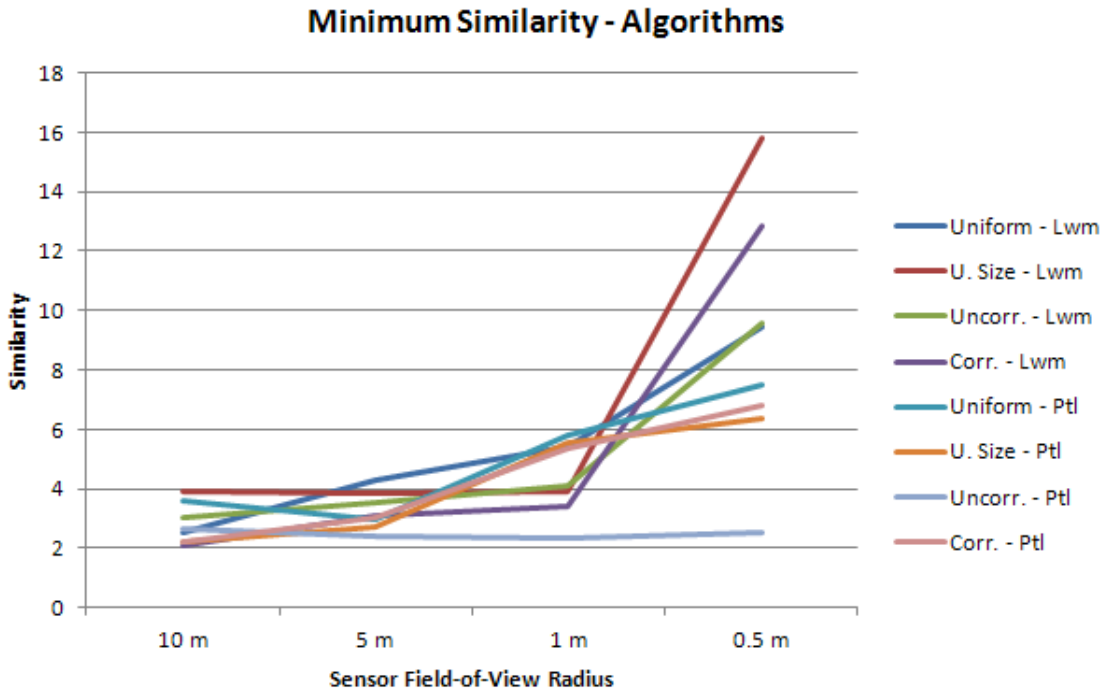


Figure 54. Minimum similarity measures for the reallocation algorithms and characteristic trajectories. Note that the overall similarity is worse (greater) compared to the data set used to generate it, in Figure 44, but the plot structure is similar.

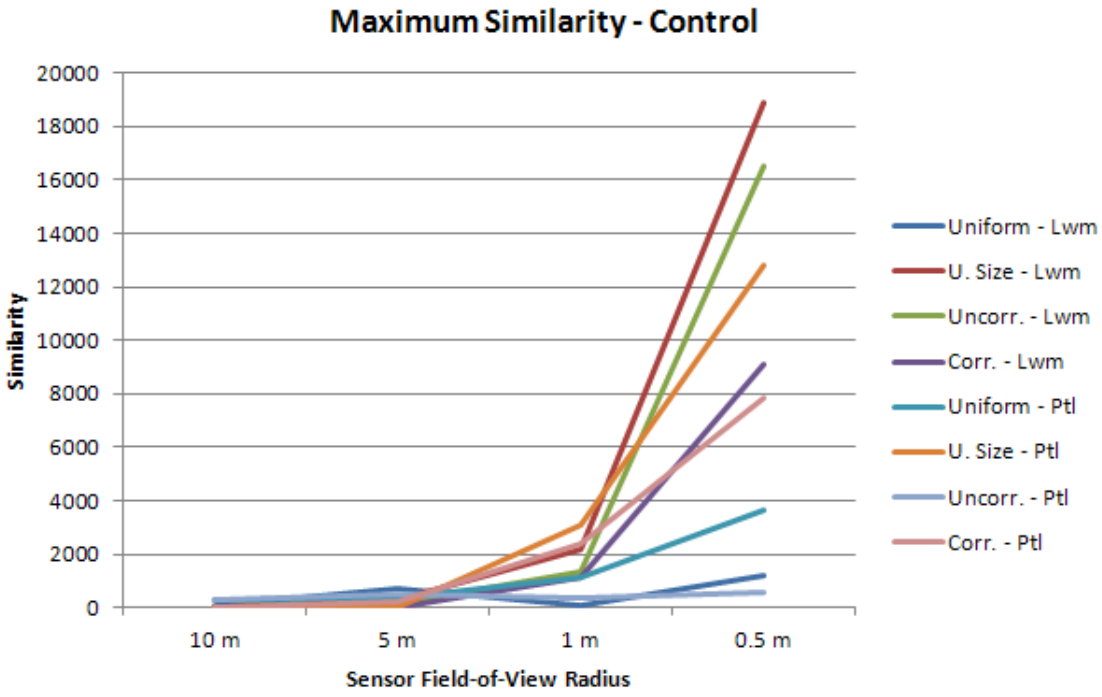


Figure 55. Maximum similarity measures for the control scenario and characteristic trajectories. Note that the overall similarity is worse (greater) compared to the data set used to generate it, in Figure 45, but the plot structure is similar.

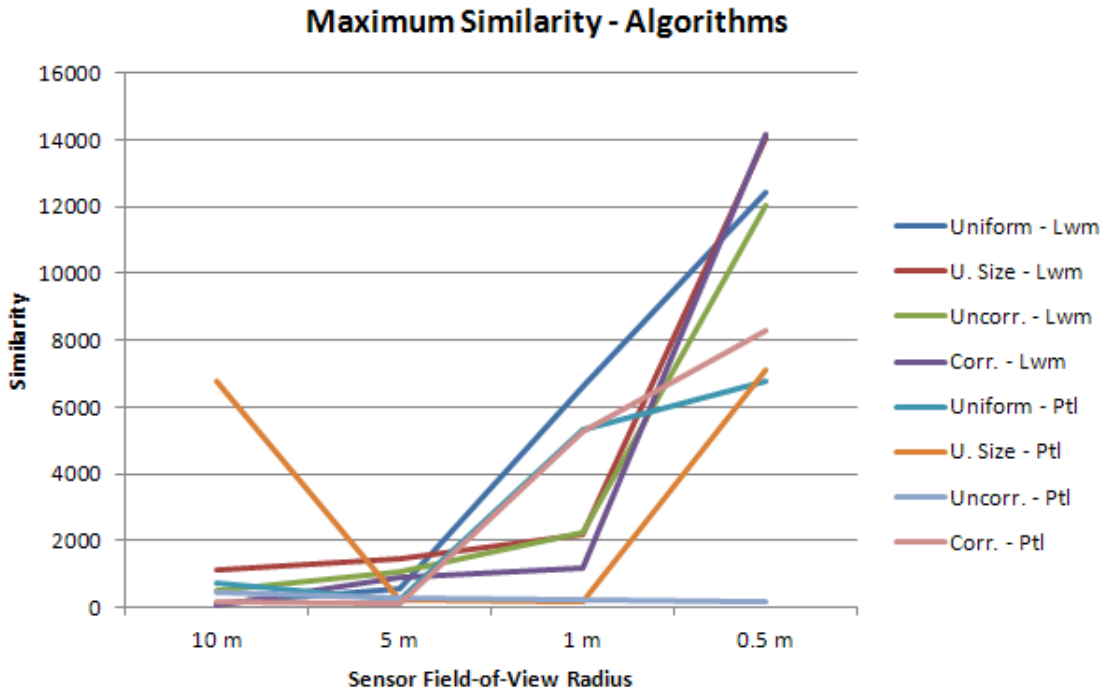


Figure 56. Maximum similarity measures for the reallocation algorithms and characteristic trajectories. Note that the overall similarity is worse (greater) compared to the data set used to generate it, in Figure 46, but the plot structure is nearly identical.

6.3 Target Coverage Evaluation

While a model may be able to cover present and future data sets effectively, as shown in the previous section, if the agents participating in a mission cannot efficiently cover the majority of the targets, then the algorithm loses some value. Hence, along with data coverage, *target coverage* will also be evaluated.

In this section, the resulting model created from a data set will be assessed to determine how well it improves future data collection such that target coverage and acquisition time is improved by reshaping the region of interest $\mathcal{S}$ using the model. Target coverage over different scenarios will be studied in detail. The base scenarios and their variants used to evaluate target coverage will be identical to the ones defined in Section 6.2.2, hence their definitions will not be repeated in this section. These scenarios will include the use of both

Table 24. Model similarity - Control, lawnmower pattern, characteristic data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	2.24	0.908	120	189	2.24	0.735
Uniform	5 m	5.52	1.64	708	405	2.51	0.523
Uniform	1 m	3.52	1.16	69.5	47	2.07	0.342
Uniform	0.5 m	3.36	1.23	1200	755	2.31	0.411
U. Size	10 m	1.64	0.752	35.6	18	2.42	0.647
U. Size	5 m	2.75	1.27	267	350	2.29	0.692
U. Size	1 m	3.99	2.38	2180	1380	2.18	0.804
U. Size	0.5 m	12.5	9.31	18900	12500	2.27	0.881
Uncorr.	10 m	1.58	0.709	37.3	15.9	2.41	0.645
Uncorr.	5 m	2.69	1.31	176	218	2.29	0.696
Uncorr.	1 m	3.58	1.74	1320	1030	2.21	0.833
Uncorr.	0.5 m	11.9	8.08	16500	8010	2.29	0.888
Corr.	10 m	1.93	0.955	32.1	16.2	2.41	0.649
Corr.	5 m	2.46	1.08	48.8	23.6	2.37	0.699
Corr.	1 m	3.48	1.74	1110	876	2.04	0.755
Corr.	0.5 m	7.78	5.0248	9110	5150	2.10	0.861

Table 25. Model similarity - Control, patroller pattern, characteristic data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	3.83	1.62	301	162	1.77	0.482
Uniform	5 m	2.73	1.27	393	531	1.13	0.576
Uniform	1 m	4.70	1.73	1150	717	1.38	0.374
Uniform	0.5 m	3.80	1.43	3630	1300	1.67	0.307
U. Size	10 m	2.05	1.17	45.7	28.7	1.99	0.793
U. Size	5 m	3.04	2.14	43	45.5	2.09	0.867
U. Size	1 m	5.08	3.16	3080	2680	2.09	0.854
U. Size	0.5 m	6.99	4.11	12800	8470	2.16	0.915
Uncorr.	10 m	2.40	1.28	279	466	2.09	0.811
Uncorr.	5 m	2.62	1.64	514	946	2.05	0.793
Uncorr.	1 m	2.91	1.79	403	851	2.08	0.810
Uncorr.	0.5 m	2.63	1.42	550	926	2.13	0.833
Corr.	10 m	2.05	1.08	44.6	17.7	1.91	0.826
Corr.	5 m	3.67	2.69	198	342	2.07	0.853
Corr.	1 m	5.69	4.03	2370	2220	2.11	0.892
Corr.	0.5 m	6.09	4.15	7830	4870	2.16	0.952

Table 26. Model similarity - Using reallocation, lawnmower pattern, characteristic data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	2.56	1.11	154	329	1.95	0.714
Uniform	5 m	4.26	1.51	597	581	2.00	0.604
Uniform	1 m	5.43	2.82	6590	7580	2.27	0.712
Uniform	0.5 m	9.44	2.87	12400	5210	2.48	0.490
U. Size	10 m	3.94	1.77	1160	855	2.43	0.680
U. Size	5 m	3.87	1.91	1470	865	2.31	0.736
U. Size	1 m	3.89	2.06	2202	1940	2.18	0.801
U. Size	0.5 m	15.8	12.3	14100	8750	2.17	0.828
Uncorr.	10 m	3.04	1.66	515	579	2.41	0.669
Uncorr.	5 m	3.52	1.76	1100	892	2.29	0.706
Uncorr.	1 m	4.11	2.34	2240	1940	2.19	0.804
Uncorr.	0.5 m	9.57	5.64	12100	6270	2.27	0.893
Corr.	10 m	2.12	0.998	98.6	55.3	2.44	0.663
Corr.	5 m	3.11	1.89	934	1410	2.38	0.727
Corr.	1 m	3.42	1.69	1180	922	2.13	0.79
Corr.	0.5 m	12.9	9.58	14200	6560	2.12	0.836

Table 27. Model similarity - Using reallocation, patroller pattern, characteristic data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	3.58	1.39	744	670	1.84	0.449
Uniform	5 m	2.98	1.22	211	329	1.18	0.554
Uniform	1 m	5.83	3.6	5340	2710	1.53	0.448
Uniform	0.5 m	7.51	5.21	6790	2703	1.72	0.371
U. Size	10 m	2.22	1.19	220	264	1.98	0.783
U. Size	5 m	2.71	1.85	155	362	1.93	0.821
U. Size	1 m	5.54	3.96	7110	6540	2.13	0.913
U. Size	0.5 m	6.38	4.44	8280	5690	2.16	0.926
Uncorr.	10 m	2.64	1.49	484	682	2.08	0.813
Uncorr.	5 m	2.41	1.32	292	405	2.07	0.814
Uncorr.	1 m	2.34	1.29	254	374	2.09	0.827
Uncorr.	0.5 m	2.52	1.41	174	480	2.08	0.809
Corr.	10 m	2.23	1.22	174	338	1.96	0.797
Corr.	5 m	3.04	2.01	116	198	2.02	0.84
Corr.	1 m	5.39	3.6	5300	4570	2.17	0.917
Corr.	0.5 m	6.79	4.29	8303	6610	2.20	0.958

Table 28. Model similarity - Fixed, characteristic data set.

Scenario	Sensor FOV	μ min D_s	σ min D_s	μ max D_s	σ max D_s	μ $D_{s,ovr}$	σ $D_{s,ovr}$
Uniform	10 m	12.1	6.06	3720	2550	1.49	0.721
Uniform	5 m	N/A	N/A	N/A	N/A	N/A	N/A
Uniform	1 m	N/A	N/A	N/A	N/A	N/A	N/A
Uniform	0.5 m	N/A	N/A	N/A	N/A	N/A	N/A
Uncorr.	10 m	3.55	2.01	542	672	2.13	1.1
Uncorr.	5 m	10.7	9.11	4290	3102	2.21	0.930
Uncorr.	1 m	16.1	18.6	40400	41900	2.20	0.935
Uncorr.	0.5 m	25.3	29.1	21700	15900	1.92	0.740

mobile and fixed agents.

6.3.1 Overview

Formally, target coverage C is defined with respect to the total time T as follows:

$$C(T) = \frac{N_T(T)}{N_{total}(T)}, \tag{55}$$

where $N_T(T)$ is the number of unique targets acquired by all sensors over the time T and $N_{total}(T)$ is the number of targets emitted by all ablating sources over time period T. That is,

$$N_{total}(T) = \sum_{i=1}^{l} N_i(T). \tag{56}$$

$C(T)$ as a measure of success may be compared to the measure of success of the general *area coverage* problem, where the measure of success is the percentage of the region that is covered by a mobile sensor. $C(T)$ may also be reworded as the following: for a model to successfully capture the ice ablation activity near a glacier, it must identify the maximum number of target streams $B_i(t)$ that pass through the subdomain of $\mathcal{S}$ formed by the model Q. A *miss rate* may be defined in terms of this function; *i.e.*, miss rate is equal to 1 - $C(T)$.

The hypothesis to be tested is that using the modeling methodology, which provides information on where to acquire target measurements and where the original target streams are located, provides equivalent or better average target coverage to a standard scanning

solution. Such a scanning solution is the restriction of mobile sensors to their own, fixed-size cells. Improving target coverage by limiting spatial coverage is in fact a difficult task. Such improvement depends on how the data to build the model is collected, similar to how spatial diversity will improve data coverage. Equivalent coverage indicates that using either scanning technique is sufficient. As the model evolves from the measurements, whether target coverage is retained or improved will be investigated. Target coverage for a standard scanning solution, *i.e.*, the fixed-cell approach, should be fixed for a given scenario.

6.3.2 Simulation Results

As stated previously, the same scenarios used to study data coverage in Section 6.2 are used to examine target coverage. The resulting target coverage for each scenario configuration for varying sensor fields of view and search patterns are given in Table 29 and Table 30. Table 31 provides target coverage for the two fixed sensor scenarios: uniform target movement and uncorrelated target movement. $\mu\ C$ is the average target coverage, while $\sigma\ C$ is the target coverage standard deviation.

To verify the statistical significance of the simulation results for target coverage, a one-tailed t-test with null hypothesis mean 0.5, 29 degrees of freedom, and a significance level of $p < 0.01$ was conducted. For this case, the hypothesis mean was chosen on the basis that the lowest tolerable target coverage is 50%; *i.e.*, which targets are covered is governed by what is effectively the result of a fair coin toss. The t-scores and p-values for each scenario are shown in Tables 32, 33, and 34. With respect to the significance level, the results are statistically significant except in the following cases, which reflect anomalous target coverage behavior which will be further explained:

- The uncorrelated velocity scenario when using the patroller pattern when either in the control or reallocation cases.
- When using reallocation and the patroller pattern in the 10 m field-of-view case and the lawnmower pattern in the 5 m field-of-view case for the uniform scenario.

Table 29. Target coverage summary - Control.

Scenario	Sensor FOV	Lwm. μ C	Lwm. σ C	Ptl. μ C	Ptl. σ C
Uniform	10 m	0.829	0.0547	0.848	0.0481
Uniform	5 m	0.724	0.0249	0.830	0.0410
Uniform	1 m	0.242	0.0281	0.351	0.0298
Uniform	0.5 m	0.107	0.0162	0.187	0.0285
U. Size	10 m	0.850	0.0284	0.828	0.0530
U. Size	5 m	0.591	0.0175	0.750	0.0364
U. Size	1 m	0.133	0.0181	0.204	0.0238
U. Size	0.5 m	0.0592	0.0174	0.098	0.0214
Uncorr.	10 m	0.859	0.0390	0.418	0.290
Uncorr.	5 m	0.586	0.0187	0.462	0.304
Uncorr.	1 m	0.127	0.0171	0.495	0.296
Uncorr.	0.5 m	0.0587	0.0200	0.421	0.299
Corr.	10 m	0.869	0.0383	0.831	0.0486
Corr.	5 m	0.640	0.0215	0.752	0.0402
Corr.	1 m	0.148	0.0143	0.218	0.0281
Corr.	0.5 m	0.0725	0.0157	0.108	0.0181

- When using reallocation in the uniform size and uncorrelated velocity cases with a 10 m field-of-view and the lawnmower search pattern.
- When using reallocation in the correlated velocity case with a 5 m field-of-view and the lawnmower search pattern.

From an initial examination of the raw data, there are some overall observations that can be made. These observations are as follows:

1. The target coverage for the fixed-cell approach is greater than that of the approach using the reallocation algorithms.
2. The standard deviations for the reallocation algorithms are high when compared to the fixed-cell approach.
3. When the sensor field-of-view radius falls below a certain value, in this case 5 m, either approach has equivalent or near equivalent performance.

Table 30. Target coverage summary - Reallocation.

Scenario	Sensor FOV	Lwm. μ C	Lwm. σ C	Ptl. μ C	Ptl. σ C
Uniform	10 m	0.662	0.0850	0.485	0.322
Uniform	5 m	0.516	0.0729	0.688	0.226
Uniform	1 m	0.243	0.0463	0.353	0.0392
Uniform	0.5 m	0.115	0.0182	0.197	0.0257
U. Size	10 m	0.455	0.306	0.750	0.0944
U. Size	5 m	0.344	0.212	0.597	0.0830
U. Size	1 m	0.125	0.0145	0.198	0.0294
U. Size	0.5 m	0.0598	0.0109	0.106	0.0192
Uncorr.	10 m	0.436	0.269	0.447	0.311
Uncorr.	5 m	0.361	0.213	0.424	0.310
Uncorr.	1 m	0.119	0.0245	0.432	0.285
Uncorr.	0.5 m	0.0618	0.0158	0.425	0.287
Corr.	10 m	0.707	0.0897	0.741	0.0892
Corr.	5 m	0.529	0.0970	0.695	0.0931
Corr.	1 m	0.149	0.0215	0.220	0.0311
Corr.	0.5 m	0.0678	0.0172	0.109	0.0186

Table 31. Target coverage summary - Fixed.

Scenario	Sensor FOV	μ C	σ C
Uniform	10 m	0.725	0.0161
Uniform	5 m	0.250	0.000913
Uniform	1 m	N/A	N/A
Uniform	0.5 m	N/A	N/A
Uncorr.	10 m	0.752	0.0166
Uncorr.	5 m	0.308	0.00240
Uncorr.	1 m	0.055	0
Uncorr.	0.5 m	0.02	0

Table 32. Target coverage p-values - Control.

Scenario	Sensor FOV	Lwm. C t-score	Lwm. C p-value	Patrol C t-score	Patrol C p-value
Uniform	10 m	32.9	6.93E-25	39.6	3.02E-27
Uniform	5 m	49.2	4.91E-30	44.1	1.22E-28
Uniform	1 m	-50.2	2.59E-30	-27.3	1.53E-22
Uniform	0.5 m	-132	6.50E-43	-60.0	1.32E-32
U. Size	10 m	67.5	3.94E-34	33.8	2.95E-25
U. Size	5 m	28.6	4.06E-23	37.6	1.33E-26
U. Size	1 m	-111	1.36E-40	-68.1	3.07E-34
U. Size	0.5 m	-138	1.87E-43	-102	1.47E-39
Uncorr.	10 m	50.4	2.34E-30	-1.55	0.119
Uncorr.	5 m	25.2	1.58E-21	-0.68	0.310
Uncorr.	1 m	-119	1.48E-41	-0.08	0.394
Uncorr.	0.5 m	-120	1.09E-41	-1.44	0.140
Corr.	10 m	52.7	6.36E-31	37.33	1.71E-26
Corr.	5 m	35.6	6.80E-26	34.38	1.92E-25
Corr.	1 m	-134	4.80E-43	-54.8	2.00E-31
Corr.	0.5 m	-148	2.25E-44	-118	2.07E-41

Table 33. Target coverage p-values - Reallocation.

Scenario	Sensor FOV	Lwm. C t-score	Lwm. C p-value	Patrol C t-score	Patrol C p-value
Uniform	10 m	10.4	2.92E-11	-0.262	0.382
Uniform	5 m	1.17	0.19	4.56	0.000117
Uniform	1 m	-30.4	6.56E-24	-20.5	5.54E-19
Uniform	0.5 m	-116	3.61E-41	-64.5	1.55E-33
U. Size	10 m	-0.808	0.28	14.5	7.11E-15
U. Size	5 m	-4.02	0.000513	6.43	6.71E-07
U. Size	1 m	-141	9.80E-44	-56.3	8.67E-32
U. Size	0.5 m	-220	1.74E-49	-112	1.02E-40
Uncorr.	10 m	-1.31	0.166	-0.928	0.255
Uncorr.	5 m	-3.57	0.00165	-1.33	0.161
Uncorr.	1 m	-85.1	3.97E-37	-1.30	0.168
Uncorr.	0.5 m	-152	1.16E-44	-1.43	0.141
Corr.	10 m	12.6	2.43E-13	14.8	3.96E-15
Corr.	5 m	1.61	0.109	11.4	2.88E-12
Corr.	1 m	-89.5	8.90E-38	-49.2	4.93E-30
Corr.	0.5 m	-137	2.32E-43	-115	4.97E-41

Table 34. Target coverage p-values - Fixed.

Scenario	Sensor FOV	*C* t-score	*C* p-value
Uniform	10 m	76.7	8.91E-36
Uniform	5 m	-1500	1.75E-74
Uniform	1 m	N/A	N/A
Uniform	0.5 m	N/A	N/A
Uncorr.	10 m	83.2	7.97E-37
Uncorr.	5 m	-438	1.94E-58
Uncorr.	1 m	N/A	N/A
Uncorr.	0.5 m	N/A	N/A

The first observation indicates that there is the presence of a trade-off in terms of target coverage and reducing target acquisition time, as will be seen in Section 6.4. The fixed-cell approaches provide a greater target coverage, and indeed, a relatively stable target coverage as indicated by the standard deviations and postulated in Section 6.3.1. This ties into the second observation: although on the average the coverage is lower, there are instances where the algorithms provide equivalent performance on the upper side when compared to the fixed-cell approach. This suggests that, for the controller configuration used in these tests, the model evolves to cover targets in a reactive manner, losing target coverage until targets that have been ablated from other ablation regions drift into the new search regions generated by the model.

Given the results of using the modeling and reallocation algorithms, when choosing how to use the model to reallocate sensors and to do it in a manner that improves *overall* performance, how target coverage is affected must be taken into account external to the reassignment of agents to new regions, even though target coverage in the sense of this section might not be directly measurable. This issue with respect to coverage also suggests that there is an optimum, where both target coverage and acquisition time have optimal values with respect to a given observation scenario.

The third observation can be clarified by plotting the means of the target coverage; these plots are given in Figures 57, 58, 59, and 60.

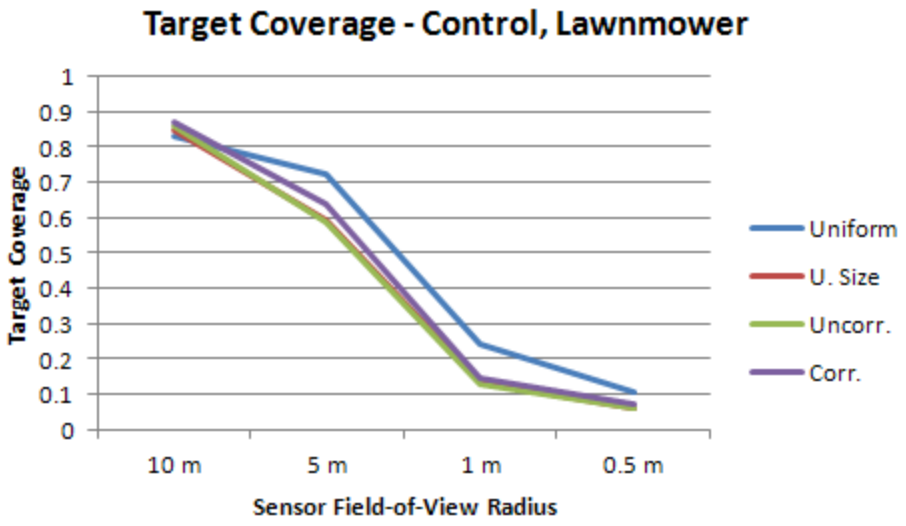


Figure 57. Target coverage for the lawnmower pattern control scenarios. The performance correlates to the sensor field-of-view.

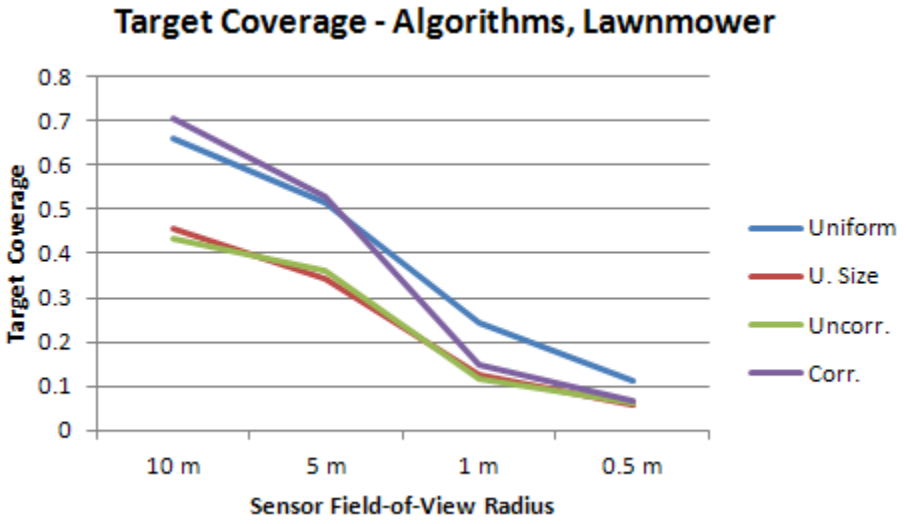


Figure 58. Target coverage for the lawnmower pattern scenarios using reallocation algorithms. Coverage is smaller in magnitude than in the control case, as in Figure 57.

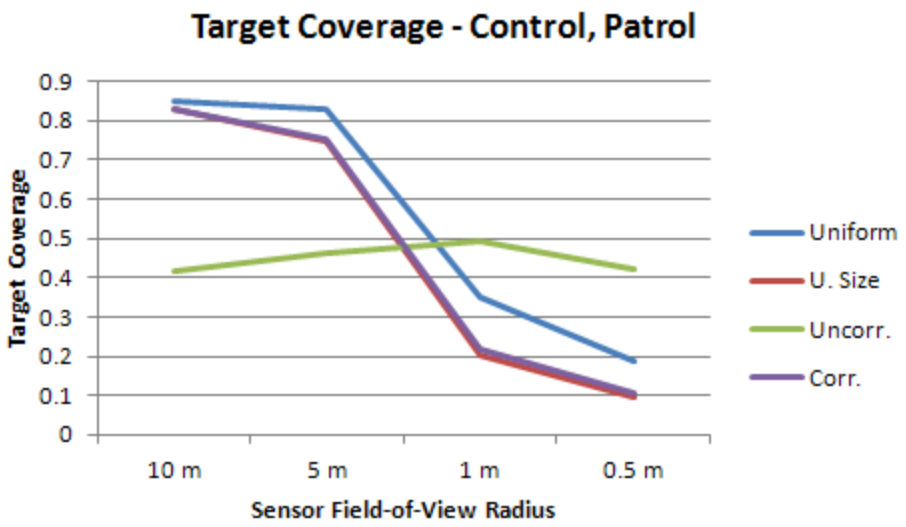


Figure 59. Target coverage for the patroller pattern control scenarios. The performance is similar to that of the lawnmower pattern in the control case illustrated in Figure 57, except in the uncorrelated case.

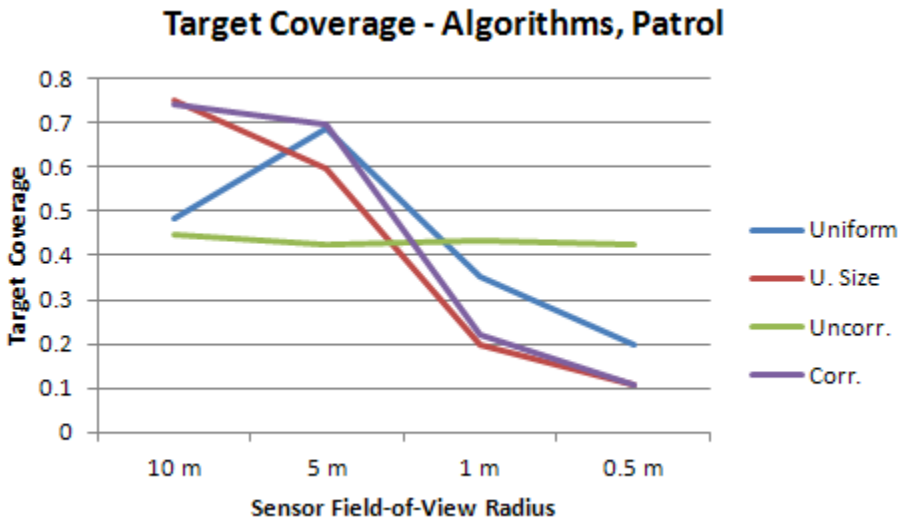


Figure 60. Target coverage for the patroller pattern scenarios using reallocation algorithms. The coverage is smaller in magnitude than in the control case, but the same behavior for the uncorrelated scenario is present as in Figure 59.

For the lawnmower search pattern, the trends between the control (Figure 57) and the

algorithms (Figure 58) are very similar, although the lines for the algorithms are shifted downward as a result of the acquisition time-target coverage trade-off. Specifically, the coverage is much lower for the uncorrelated velocity and uniform size scenarios. This result can be interpreted as follows: the target size, if the target can be acquired, is not as much of a deterrent to performance as target movement, since the uniform size scenario uses the same type of uncorrelated velocity variance as the uncorrelated velocity scenario. In both cases, the target coverage drops significantly between the 5 m and 1 m scenarios, reaching equivalence. This equivalence indicates that the target coverage is bounded once the sensor field-of-view radius approaches these sizes; the small magnitudes of the standard deviations for these cases makes it more clear.

Similar statements may be made for the patroller pattern cases as shown in Figures 59 and 60 in terms of overall trend similarity as the sensor field-of-view radius is modified. The overall coverage is greater in magnitude for the control with the two larger sensor field-of-view radii, but it decreases quickly to the same lower limits as that for the lawnmower search pattern. However, for the patrollers, there is the interesting case of the uncorrelated velocity variance scenarios. In both the control and reallocation cases, the target coverage is very similar and nearly constant across all four scenario variants, which is a contrast to the similarity of this scenario with the uniform size scenario for the lawnmower search pattern. This difference may be attributable to the spread of the targets as generated by this scenario when compared to the correlated velocity scenario, which tracks closely with the uniform size scenario. Targets in the correlated velocity scenario may have a tighter spread than that of the uncorrelated velocity scenario depending on the correlation between adjacent ablating source regions, providing comparable performance to the uniform velocity scenario. A wider spread for a patroller, which has reduced coverage properties than the lawnmower pattern, can work to its disadvantage. If a target is altogether missed or does not approach the sensor's field-of-view as the sensor is moving toward the target's future location, then there is a high likelihood that the patroller pattern will not acquire the target

at all. Hence, for a patrolling pattern, it is almost required that the agent's sensor has a large field-of-view.

Finally, the target coverage for the fixed sensors is interesting when compared to the target coverage using mobile sensors. With the low-magnitude standard deviations in the target coverage, it can be stated that for fixed sensors, the coverage is effectively constant, which should be expected. In two cases, the 1 m and 0.5 m cases for uniform target velocities, sufficient targets could not be acquired to actually compute target coverage. Compared to the uncorrelated target velocity case, it is likely that the target coverage would be no higher than approximately 5.5% even in the uniform case. In all cases, for both the fixed-cell approach and using the modeling and reallocation algorithms, there is an improvement in target coverage in comparison to the fixed-sensor approach.

It should be noted that although the resulting average target coverage for the algorithms is lower overall, in a real-world scenario, that does not mean that there is a significant issue to overcome. Specifically, the scenarios used to test target coverage were intended to provide a significant challenge to the algorithms in terms of target coverage. To make this clearer, the contrast of the target acquisition time with target coverage is shown in Figures 61, 62, 63, and 64, where the acquisition times for both the control and algorithm scenarios using the lawnmower search pattern are plotted. The fixed sensor and patroller cases are not shown because in several of the test scenarios, there was insufficient data to compute appropriate acquisition times as a result of deficiencies in the ability of those methods to acquire targets.

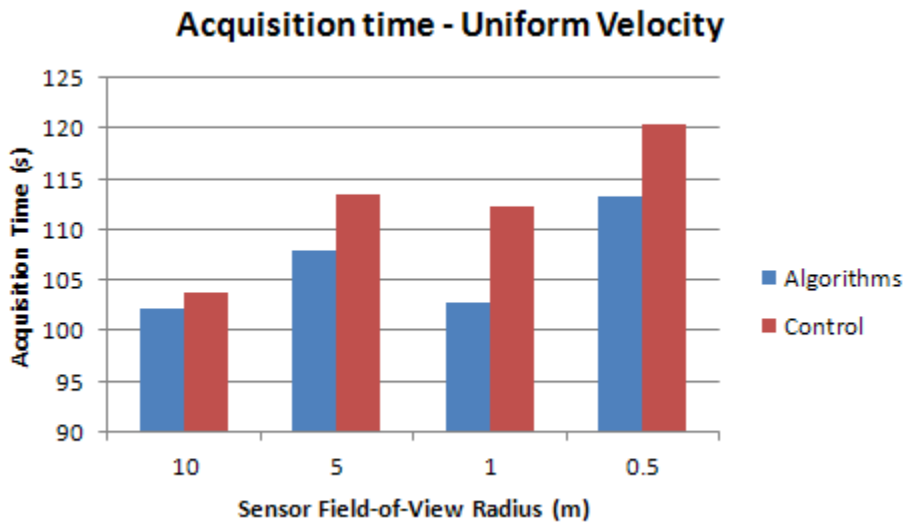


Figure 61. Acquisition times for the uniform target velocity scenario. Note the overall improvement in the acquisition time when using reallocation of resources.

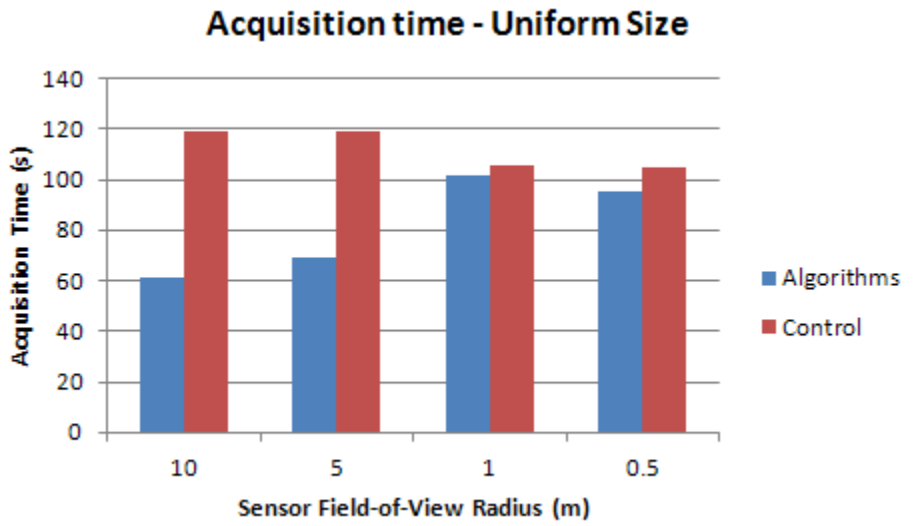


Figure 62. Acquisition times for the uniform target size scenario. There is significant improvement in the acquisition time when using reallocation, except in the small sensor field-of-view cases, corresponding to the coverage results.

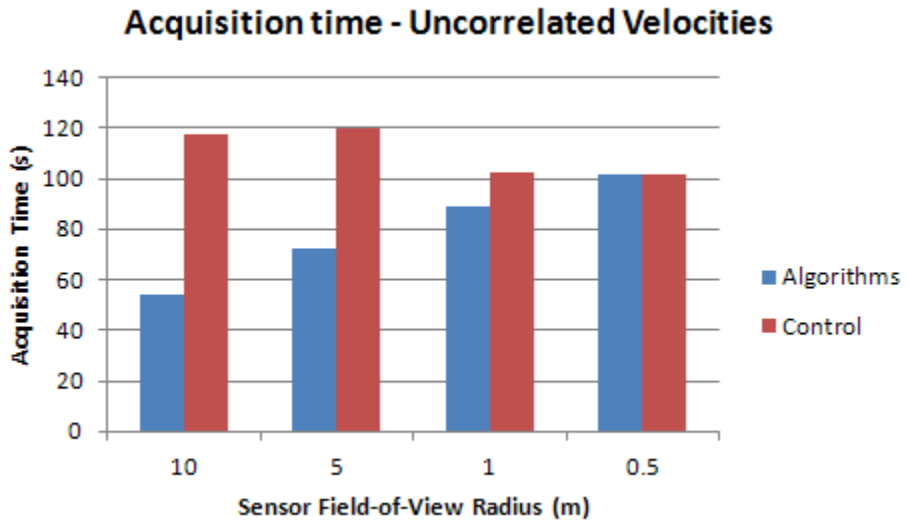


Figure 63. Acquisition times for the uncorrelated velocity scenario. There is no improvement in the acquisition time for the smaller sensor fields-of-view, while the improvement is present in the greater field-of-view radii for the sensor reallocation algorithms.

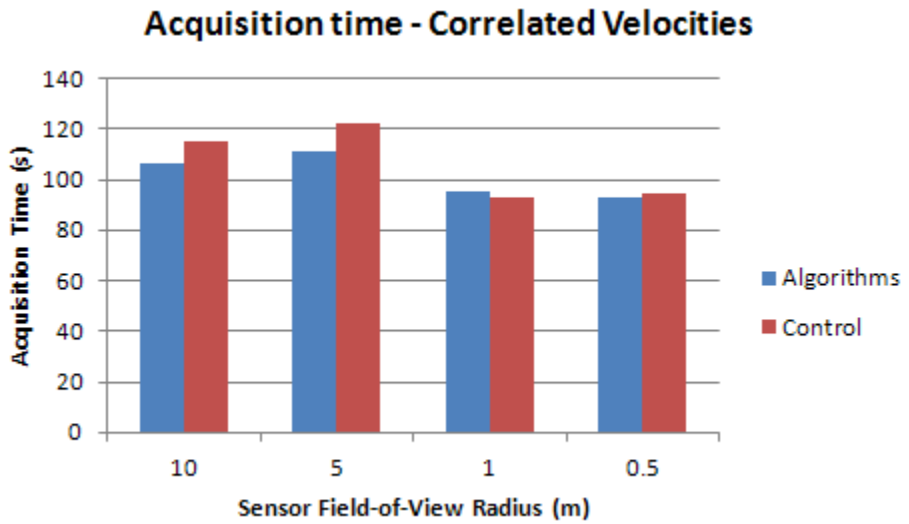


Figure 64. Acquisition times for the correlated target velocity scenario. Note that there is improvement when using the reallocation algorithms, but the improvement is not as drastic as in the previous scenarios.

In almost all cases, there is an improvement when using the modeling and reallocation algorithms over the control methodology for performing target acquisition. The cases where there is a lack of improvement or the resulting acquisition times are very similar are in the 1 m and 0.5 m sensor field-of-view radius cases, which is to be expected, considering target coverage performance for these cases. Additionally, the correlated target velocity case proved to be a more difficult case as well, with the difference between the control and algorithm scenarios not as significant.

It should also be noted that real icebergs might not necessarily drift in such a structured manner, given the dynamics of ocean currents that are not modeled in this ablating source simulation. That is, current vortices can cause clusters of icebergs to become trapped in a smaller subregion of the region of interest $\mathcal{S}$ for a significant amount of time, as suggested by the results of the IIP data analysis in Section 3.5. This behavior would make acquiring the icebergs less difficult and increase target coverage.

6.4 Full-Scale Simulation

Given the results of the previous studies described in the previous sections, a full-scale simulation incorporating the modeling and assignment techniques from Chapters 3 and 4 was conducted [58, 87]. This full-scale simulation usd the same simulation environment as the pilot study: the same controller was used, except that the additional target assignment and modeling options were activated. Unlike the target coverage and data coverage studies, the ablating sources U would also vary substantially to model the starting and stopping of the target streams $B_i(t)$ generated by the ablating sources u_i.

The overall simulation parameters are similar to that of the pilot study summarized in Section 6.1 and the scenarios used in the coverage evaluation of Section 6.3. The same target source parameters and robot parameters in the pilot simulation were used to provide an appreciable extension to the study. The main difference was the number of agents used in each of the studies. The "control" simulation, which did not use any of the modeling or

assignment techniques, is essentially the three-agent solution from the pilot study. As for the solution that uses modeling and assignment, a patroller agent is added to the scanning agents, which is used to perform the patrolling task as described in Section 5.1.3 as part of aiding an appropriate *a-priori* mixture model. This agent switches to a lawnmower pattern once the first region assignment has been made as the *a-priori* model will have been obtained; while this seems to be an additional advantage for the modeling and assignment solution as compared to the three-agent control solution, as shown in the pilot study, the most coverage that can be obtained for the given, overall scenario is obtainable with three agents.

The different simulation scenarios based on the overall scenario are summarized in Table 35. In each case, the number of target streams $B_i(t)$ is varied and the amount of ablation capacity of each of the streams is varied. Varying the capacities of each of the sources allows for fully exercising the capabilities of target modeling and stream detection. A stream with a regular-sized capacity has an ablation capacity of 1000 tons; smaller streams have capacities of 200 tons. While these capacities are smaller than the potential capacity of a real ablation source, they are comparable for the circumstances of this particular test. Each scenario was run for 3000 simulation frames with a frame rate of 25 Hz, with accelerated ablation rates as in the pilot simulation. Thirty trials of each simulation were run.

The results of the control simulation are summarized in Table 36 and the results of the simulation incorporating modeling and assignment are summarized in Table 37.

Some general observations can be made about the results in Table 37. First, the agent-specific target coverage $C_A(T)$ remains relatively constant across all of the scenarios, regardless of the algorithms used. Second, there is significant improvement in the initial acquisition time, which is the overall metric of interest. Finally, the average distance traveled remains similar across scenarios. While it can be observed from the raw metrics data that the solution using the modeling and assignment algorithms is an improvement over the control, as most of the numbers have significant drops, especially in the average acquisition

Table 35. Simulation Scenarios

Identifier	Description
four	Four target streams of equal capacity.
three	Three target streams of equal capacity.
two	Two target streams of equal capacity.
one	One target stream of equal capacity to those before.
one_small	Three target streams of equal capacity, one stream of small capacity.
two_small	Two target streams of equal capacity, two streams of small and equal capacity.
three_small	One target stream of "regular" capacity, three streams of small and equal capacity.

Table 36. Simulation metrics summary - Control.

Scenario	$C_A(T)$	Avg. T_s (s)	Avg. Model T_s (s)	Avg. Local T_s (s)	Avg. Dist. (m)
four	100%	12.8	494.13	14.36	311.65
three	100%	13.76	475.07	16.50	296.53
two	100%	14.89	453.20	15.40	292.26
one	100%	21.56	425.27	N/A	279.50
one_small	100%	13.90	415.47	16.05	307.5
two_small	100%	10.01	288.94	12.48	298.72
three_small	99.9%	12.11	212.99	12.83	300.10

Table 37. Simulation metrics summary - Modeling and Assignment.

Scenario	$C_A(T)$	Avg. T_s (s)	Avg. Model T_s (s)	Avg. Local T_s (s)	Avg. Dist. (m)
four	100%	7.30	401.73	15.85	263.05
three	100%	8.82	405.04	31.6	261.09
two	100%	6.3	326.19	16.03	235.09
one	100%	17.38	394.93	N/A	324.32
one_small	95.8%	8.15	363.88	32.15	263.14
two_small	93.6%	11.0	202.24	79.93	250.63
three_small	100%	11.7	135.06	66.69	242.96

time, some clarity can be provided by graphing the data. Figures 65, 66, 67, and 68 compare the two scenarios' average acquisition time, average global model acquisition time, average local model acquisition time, and the average distance traveled.

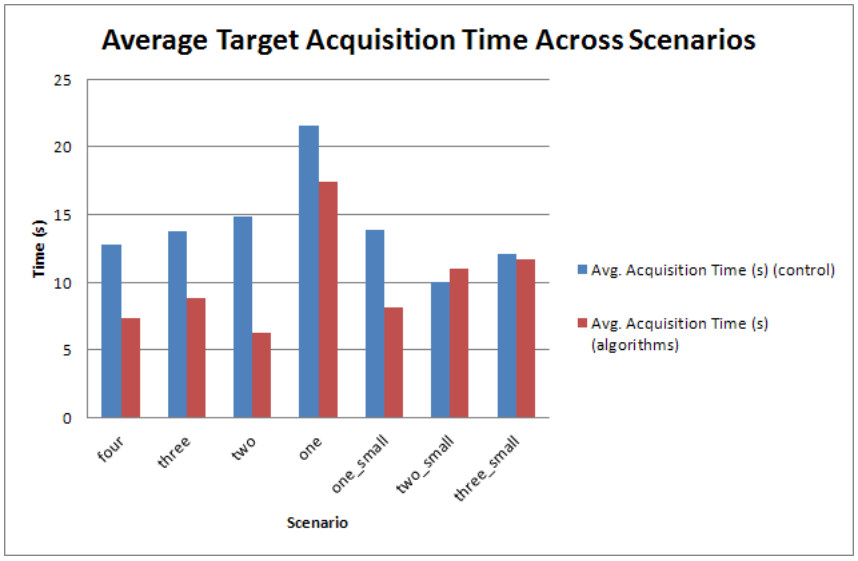


Figure 65. Average acquisition times. Note the overall improvement across the scenarios when using the reallocation algorithms.

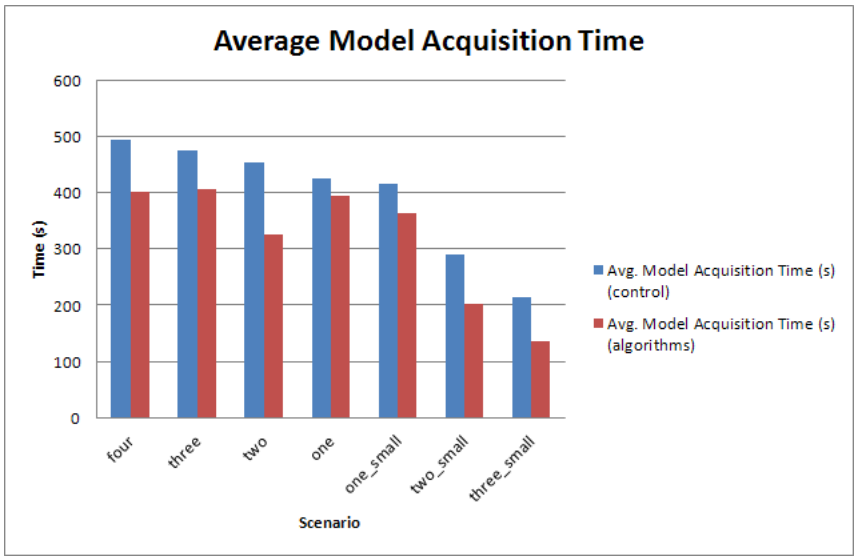


Figure 66. Average global model acquisition times. Individual global models indicate an acquisition time improvement when using the reallocation algorithms.

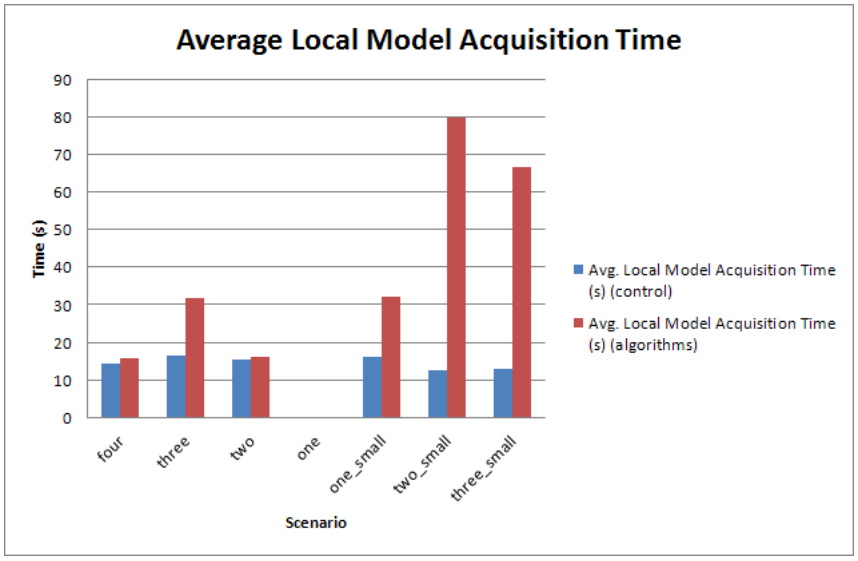


Figure 67. Average local model acquisition times. Local models do not show the same improvement, but do show the overall trend of the difficulty of obtaining target measurements as the number of target streams decreases.

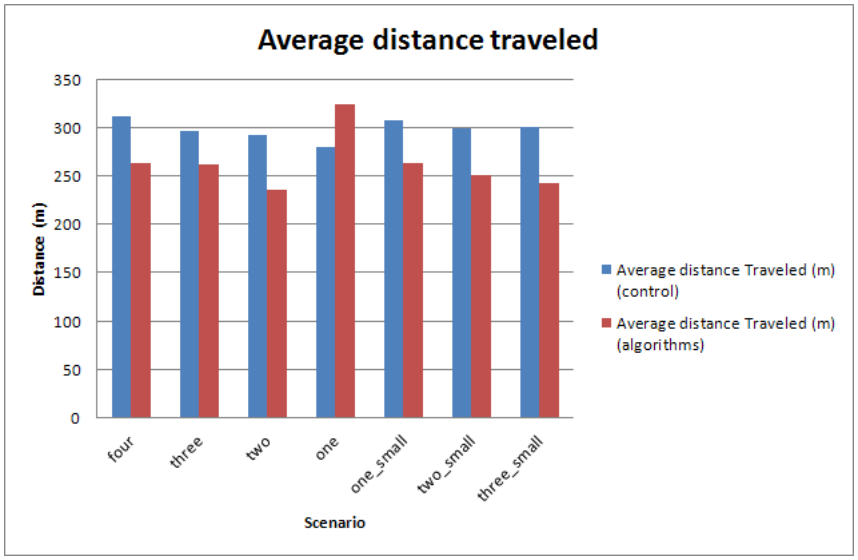


Figure 68. Average distance traveled. Note that the distance traveled has a small difference between the control and reallocation algorithms. This is indicative of obtaining better performance in terms of acquisition time when using the same amount of energy.

In almost all cases for both the control and the modeling-based methods, the agent-specific target coverage $C_A(T)$ remained fixed at 100%, which was suggested by the pilot

simulation's 100% agent-specific target coverage when using three agents. There are two specific cases where the target coverage drops: in the one and two small ablation region scenarios when using modeling and assignment. This drop is attributed to having multiple agents cover a single target stream, while a single agent remains to cover the other streams. This case does not occur in the three small stream scenario as a result of the fact that a single agent is sufficient to cover one stream; as the other streams disappear at nearly the same time, this results in "overcoverage" for the single target stream once the other streams vanish.

In Figure 65, an approximate 50% or more reduction in the initial acquisition time required when using our methodology can be observed. The cases where the behavior is not 50% or more is in the single target stream case, and the two and three small stream cases. In the single target stream case, there is a similar overcoverage case as with the three small streams case when a drop in target coverage is observed. While all of the targets are acquired, more time is required to detect targets as a result of the fact that there is only one active stream. This is the result of the fact that an agent is not constantly occupied with many target sources at a time. Hence, while several agents may be reallocated to this one stream, only so many targets exist: the acquisition time increases. Similar circumstances exist for the two and three small-stream cases, which is why the acquisition times are so similar. They are lower than the single-stream case because unlike the single-stream case, multiple streams existed at one point, allowing the agents to be occupied with quickly acquiring targets, driving down the acquisition time.

The trends in the average model-acquisition times, both local and global, are interesting. A model acquisition time, to review, is the time that a target is acquired referenced from the time that the first observation is made by any agent. The global model acquisition time is computed using the timestamps of measurements shared across agents, while the local model acquisition time is computed using the timestamps of the local measurements of each agent, and averaged across the individual local models computed by each agent.

There is improvement in the global model acquisition time when using modeling and assignment. Also, the time has a decreasing trend as target sources are removed or are of smaller capacity. This can be attributed simply to the fact that the overall number of targets was reduced, hence less time was required to acquire the possible targets. As for the local model results, the control remained flat in terms of variance overall, while there were significant changes when using modeling and assignment. The first issue to examine is the fact that there are no local results for one target stream. This result can be attributed to the fact that for many of the agents, they will not have sufficient results for one stream except for one agent or when agents are reallocated. Hence, an average cannot be computed. The significant increase in the local times for the two and three small stream cases can be attributed to the overcoverage problem: too many agents are attempting to cover too few targets.

Finally, the distance traveled in both cases remains fairly close, with some reduction when applying the algorithms. This is attributable to the shorter distance required to be traveled by the agents as they generally are allocated to smaller search regions, but the distance required to travel from search region to search region tends to offset these reductions. Hence, the difference between the control and the scenario using the algorithms is not as significant. This shows that better performance can be obtained at the same energy cost by using the algorithms.

Overall, given the metrics for performance and the differences between the simulation scenarios, clear improvement in target acquisition time can be observed when using the modeling and reassignment algorithms. This is especially true for the most important metric of interest, the average acquisition time across all agents and targets, which represents the iceberg observation framework problem's objective function. However, as noted in Section 6.3, this improvement arrives at a loss in target coverage $C(T)$, and underlines the overall suboptimality of the approach with this trade-off. Given that, with the observations regarding target coverage and acquisition time, a point at which there is an optimal balance

between these two metrics is conjectured to exist for a given scenario. However, it should also be noted that in a real-world observation mission, there are only approximations to the scenarios used to test the algorithms as a result of the variability of the ocean. Hence, while there may be an optimal point at some time in the observation mission, this optimal point is not guaranteed to exist throughout the lifetime of the ablation regions U that currently exist on a given glacier.

6.5 Hardware Results

A test was conducted using the hardware configuration described in Section 5.4. The test environment was a laboratory environment with a motion tracking system, the NaturalPoint OptiTrack system using the Motive software package, to handle localization of the individual agents, since the localization problem is not the problem of interest in this research. Appropriate software, *i.e.*, a Player driver, was written to transfer the robot positions from the motion tracking system to the controllers used to drive the individual agents. A photograph of the complete test environment is shown in Figure 69.

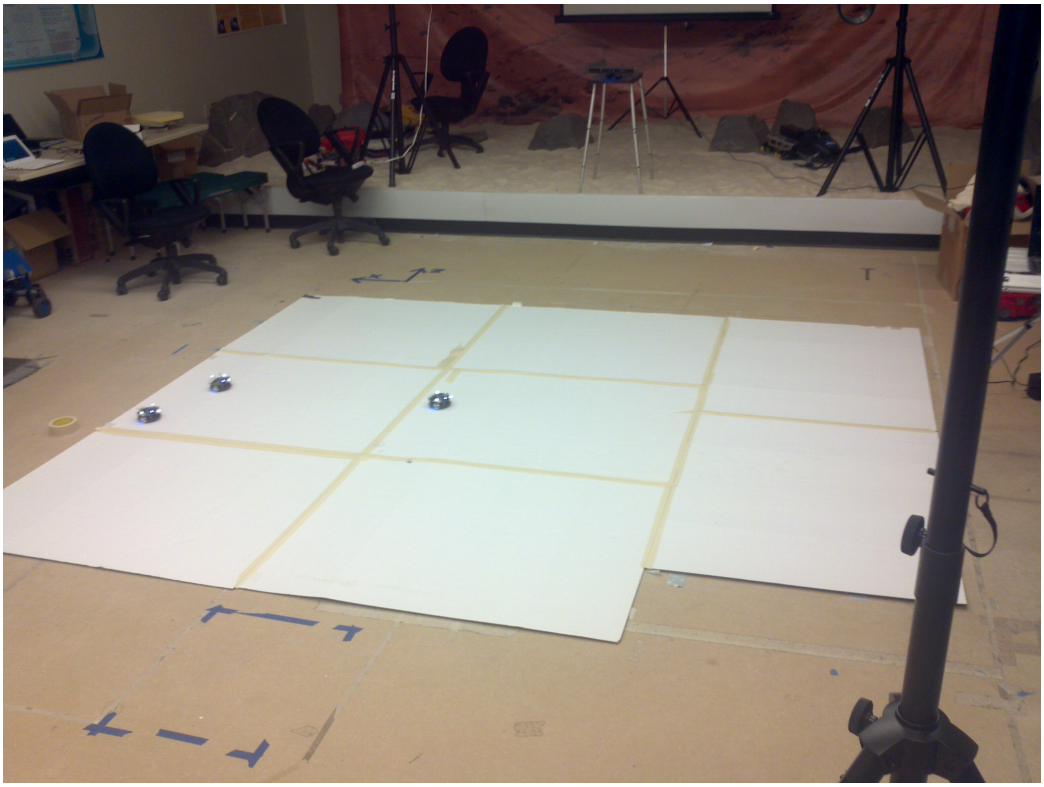


Figure 69. Lab environment for hardware experiments.

The test scenario for the hardware was a scaled-down version of the scenario used in the simulations of Sections 6.1 and 6.4. Instead of four total agents, there are only three: two agents executing a search pattern and one agent acting as the patroller agent. The default allocation is similar to that of the simulation, except that there are two cells instead of three. An illustration of this scenario is given in Figure 70. Since the purpose of the hardware test is for a validation in the real world of the algorithms on actual hardware, fewer runs (five each) were conducted than the many runs that were used to aggregate the simulation results.

Table 38. Summary of activity region parameters - Hardware Test.

Active Region	μ_{dims}	σ_{dims}	μ_{vel}	σ_{vel}
1	0.5 m x 0.5 m	0.1 m	(-0.1, 0.1) m/s	0.05 m/s
2	0.5 m x 0.5 m	0.1 m	(-0.1, 0.1) m/s	0.05 m/s
3	0.5 m x 0.5 m	0.1 m	(0, 0.1) m/s	0.05 m/s

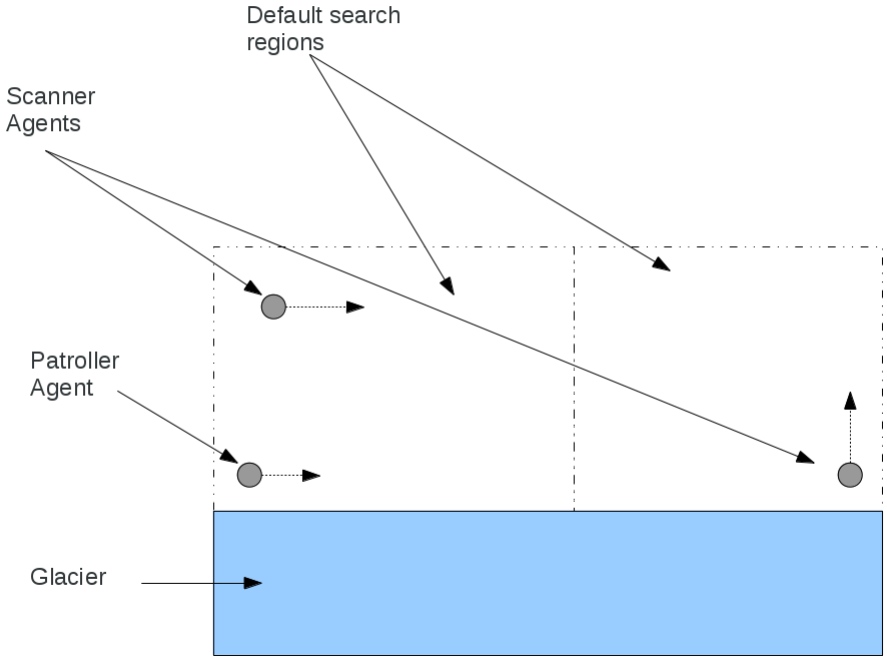


Figure 70. Initial agent allocation for hardware scenario.

The parameters for the activity regions for the hardware test are summarized in Table 38, in the manner of the activity region parameters described in Section 6.1. The virtual ice mass is centered at the OptiTrack origin, $(x, y, z) = (0, 0, 0)$, and it is allowed to generate targets in all directions. Each source had an ablation capacity of 1000 tons, with each generated target using 20 tons of this capacity and generated in 30 second intervals. The patroller agent is placed at the starting position $(x, y, z) = (-0.5, 0.1, 0)$, and the scanning agents were placed at $(x, y, z) = (-0.5, 0.2, 0)$ and $(0, 0.2, 0)$, respectively. The sensor field-of-view has a radius of 0.5 m.

Table 39. Hardware metrics summary.

Type	Coverage	Avg. T_s (s)	Avg. Model T_s (s)	Avg. Local T_s (s)	Avg. Dist. (m)
Baseline	100%	110	390	112	23
Algorithms	100%	110	390	112	23

The results of the test for the given scenario are shown in Table 39. Examining the resulting metrics determined from the hardware tests, as compared with the simulation results, very little difference exists between using the "standard" scanning approach as compared to using the modeling and reallocation methodology. Indeed, they are identical, within rounding error. Given the results for the full-scale simulation, where there was a clear distinction between the control results and results using the algorithms, the fact that the metrics are equal is likely the result of the *size* of the region of interest $\mathcal{S}$ that is being examined by the agents. Since the region of interest is much smaller compared to that of the full-scale simulations, including the distance required by the lawnmower search pattern, the target trajectories $B_i(t)$ generated by the ablation regions U will be much more closely spaced within the region of interest. Hence, for small regions of interest, such as a small coastal inlet where a smaller glacier is calving icebergs, there is little advantage to using reallocation over using fixed-size cells for scanning.

However, despite the fact that the metrics are not different for small regions of interest, the observation mission still produces a model that may be used for predicting iceberg behavior. The resulting models can be visually examined to determine how effectively the generated mixture models compare with the ablating-target sources. Figures 71, 72, and 73 show a selection of the global mixture models as perceived by the mission arbiter in the scenario. The mission arbiter, in this case, is the agent with identifier 1. In each figure, the white circles are the ablation regions, the black dots are the truth positions, and the white stars are the noisy target position measurements.

In Figures 71 and 73, it can be seen that there are three regions that correspond to

the ablation regions, specifically the target streams. Additionally, Figure 73 shows two regions of very high target-probability directly over the ablation regions, which is exactly what is desired for this modeling methodology. Figure 71 has what appears to be a "stray" component on the leftmost side, which covers the targets traveling in that direction that are not covered by the other components. Note also that in Figure 73, the measurements are not as tightly clustered as in Figure 71, providing for a greater probability near the ablation regions. For a real-world region, taking of the IIP models in Section 3.5 as an example, there is a resemblance to the models where the iceberg sightings are closer to shore, with "stronger" (components with higher probability) components near the shore, and "weaker" (components with lower probability) components further out at sea.

For Figure 72, the target streams themselves are not as clearly demarcated; most of the probability is centered atop the ablation regions, which can be compared to the activity that a real-world glacier with a very low ablation rate would produce. With respect to target streams, for determining the best regions for placing sensors, the three mixture components near the top of the plot are the best candidates, as they will capture all of the targets that float in that direction, again given that the directions of the target streams $B_i(t)$ remain constant in $\mathcal{S}$ according to the problem assumptions. For the small region of interest within the lab, if the the agents could be made smaller in size in proportion to the region of interest $\mathcal{S}$ that is being examined, then the reallocation would more likely provide the desirable results of lowered acquisition times in comparison to target coverage.

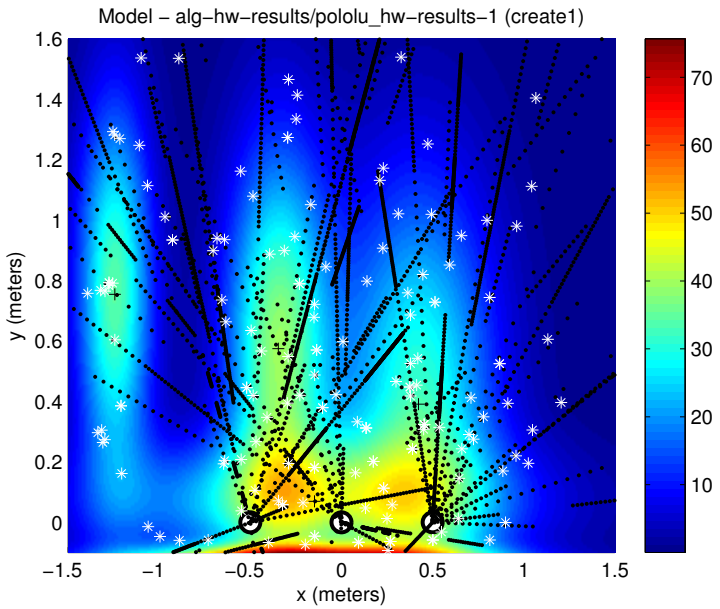


Figure 71. Hardware target model showing distinct target streams.

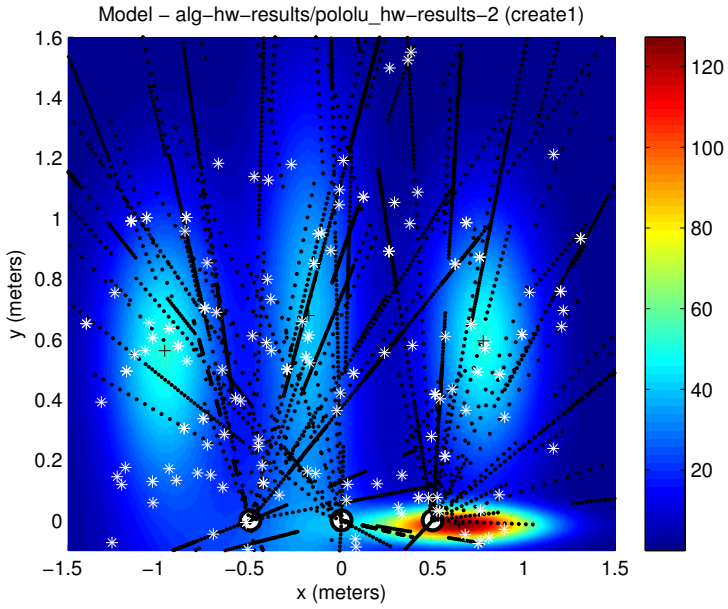


Figure 72. Hardware target model showing regions correlating to target sources.

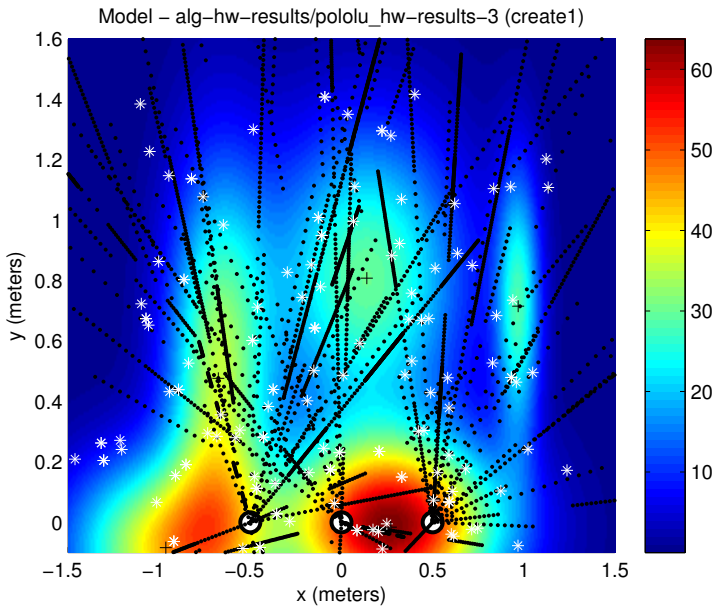


Figure 73. Hardware target model showing fewer well-defined target streams.

6.6 Summary

Given the results in this chapter, it can be said that each method of observation has its own advantages and disadvantages depending on the conditions of the area of interest $\mathcal{S}$. Table 40 shows a selection of the features of interest that can affect the choice of method for observation, "Coverage", in this case, refers to target coverage. The table includes the fixed sensor approach, which can only be improved by spending money: that is, to obtain sensors that can detect wider areas. Although the modeling and reallocation algorithms improve target acquisition time, the target coverage is inferior to that of the "standard" observation methodology. However, there are cases when this loss of coverage does not have a significant impact on situational awareness. For example, icebergs that enter the region of interest $\mathcal{S}$ and almost immediately depart at the boundary usually present no danger to operations within $\mathcal{S}$. Also, there is the case of current vortices as mentioned in Section 6.3.2. Icebergs that are caught within these vortices will persist within those regions, and provided that they do not impose on the operations within the region, they will also not present a significant threat. Finally, it should be noted that for the modeling and

Table 40. Features of each observation method.

Method	Coverage	Acquisition Time	$\mathcal{S}$ Small Area	$\mathcal{S}$ Large Area
No reallocation	×		×	
Reallocation		×	×	×
Fixed sensor			×	

reallocation techniques used in this chapter, in each case, *a-priori* data was not used in the simulations, as the ablation activity was considered to be "new" or previously unreported. This is a more difficult situation for the algorithms; with *a-priori* data, the ability of the algorithms to improve overall observation characteristics, *i.e.*, both target coverage and acquisition time, would be augmented.

As for practical recommendations as to what observation techniques should be used for a given scenario, these can be derived from the results of the simulation studies in Section 6.3, which compare target coverage and target acquisition time. The following list summarizes these recommendations based on the results:

- Use sensors with a sufficiently wide field of view. This may seem obvious, but as shown by the target coverage results, sensors with a smaller field of view require more sensors to amply cover an area and produce consistent models, which has both monetary and performance costs.

- Use fixed sensors as an adjunct to mobile sensors, but not as the primary means of observation, since the target coverage is significantly reduced with fixed sensors.

- If maximized coverage is desired, then it is best that no reallocation should be performed. Properly defined, static search regions will provide for ample overall target coverage. In addition, since coverage performance for most of the scenarios tested is effectively independent of search pattern, a patrolling or loitering pattern usually serves well for this type of target observation.

- In contrast, if the case actually occurs that the target movement is uncorrelated in

velocity, the lawnmower search pattern is a superior choice when compared to the patroller pattern for target coverage.

- If coverage is not paramount, that is, there are ablation regions ejecting new icebergs that pose no danger to a sea-based mission but there are ablation regions that will be a threat, acquisition time is the metric on which more focus should be placed. In this case, reallocation of sensor resources will improve acquisition time in all scenarios of concern.

- If the sensor resources are to be reallocated, the lawnmower search pattern is a better choice for overall target coverage performance.

CHAPTER 7

CONCLUSION

7.1 Concluding Remarks

This research addresses issues regarding the observation of targets generated from ablating sources and the development of methodologies to observe the targets more efficiently. By incorporating measurements of target position into a probabilistic model, search regions can be extracted from the model that allow for an allocation of robot resources that provides for better target acquisition. By taking into consideration the search regions in addition to how to allocate the resources, the amount of time required to obtain the initial detection on a target can be minimized. In the case of the types of targets considered in this research, *i.e.*, icebergs, minimizing the time to detect targets is integral to safe, sea-based operations in Arctic regions. The main contributions to achieving this goal are as follows: a definition of the iceberg observation problem based on an existing robotic observation problem; a probabilistic method for *in-situ* modeling of the expected locations of regions of activity on an ablating target source, so that observation resources can be retasked to these regions; a method for scoring the model based on a set of metrics defined according to the parameters that are of concern to such an observation task; and a framework for using multiple robotic agents to acquire the necessary measurements to be incorporated into the model.

The contributions are important to addressing the problem of understanding and mitigating the threat of icebergs for arctic operations. Minimizing the amount of time required to acquire these targets is important, since the types of operations that occur in these regions are performed by slow-moving ships or immobile platforms. Countermeasures of some type must be instigated as soon as a threat is detected to preserve the well-being of the crews of these missions. Satellite imagery and data products by organizations such as the IIP help to mitigate these threats, but a robotic system that is continuously monitoring

for threats and adapting to the current conditions can provide real-time situational awareness, which is important as data products are often only available on a daily or weekly basis. This research provides for such situational awareness and can provide its own data products in real-time.

As new robotic sensing platforms for Arctic exploration are developed, *e.g.*, [88], algorithms for efficiently exploring regions and gathering data will be needed, which are provided by these modeling techniques. In addition, the latest emphasis on the "nowcast" for sea ice [12] is a useful application of the models generated from the observation process, since the spread of the icebergs can be derived directly from the models. Usually, forecasts are produced by data assimilation [13] of offline data [14–16], but the methodology from this research provides for an online method of generating such a potential nowcast, using both the model and the metrics generated from the model. Therefore, the overall impact of this research is that it provides algorithms that are usable right now for improving real-time situational awareness in Arctic regions and augments existing means of observing icebergs.

7.2 Recommendations for Future Work

The logical extension to this research is to develop a full iceberg tracking system [89] that uses the model generated by the measurements gathered by the robots as part of its initialization process. A likely candidate for such a tracker would be a multiple-hypothesis tracker (MHT), similar to the system briefly described in [3].

An MHT, as opposed to other tracking systems, is a multiple target tracking system that solves the data association problem by representing a set of target tracks as a set of track hypotheses. These hypotheses are based on the possible ways that target measurements may associate with a track. Every time that a conflict between the current tracks and the current set of target observations occurs, a new set of hypotheses are generated. This method is different from other tracking systems in that usually only a single set of assignment solutions is allowed, resulting in many more sets assignment solutions that must be stored. However,

the solutions are adapted over time to produce the best possible assignment for the data, one of the primary advantages of using an MHT.

An MHT may be implemented in a variety of ways; different algorithms have been developed to solve the data association and hypothesis generation portion, while creating and updating the tracks is usually a derivative of Kalman-filter-based target tracking. A summary of several of the algorithms used to implement an MHT is given in [90].

An MHT would be an appropriate candidate for the core of an iceberg tracking system for the following reasons:

- Target classification may be difficult; multiple targets may be identified as the same target if they are tightly clustered together, which is entirely possible for small, newly calved icebergs. Target properties such as the radar cross section (RCS), size, and velocity can account for these mistakes in target identification. Using an MHT allows for eventual separation of targets with proper data management.

- Icebergs can and will break up into smaller icebergs. These new icebergs can become new hypotheses in the tracker, or a hypothesis that an iceberg has a growing probability of breaking up can be maintained.

The main issues involved in developing such a tracking system would be the issues that are often associated with the MHT and tracking systems in general:

- Data management: Currently, the entire measurement history is kept to generate the mixture models for the ablating sources. Certain tracking algorithms require storing the entire measurement history, as well [91]. These tracking algorithms are usually batch-oriented in nature. MHT-based algorithms may also be batch-oriented, and thus require that a large amount of measurement data be stored for the duration of a mission.

- Updating the tracker: As the mixture modeling method is batch-oriented, a large history of measurements is kept, and this history must be incorporated into the tracking

algorithm. Time and complexity requirements must be taken into account when updating the tracker, and some MHT-type algorithms require a fair amount of time to update. In this case, the targets will likely not have to be revisited as often as faster moving targets. As a result, the tracker updates will be more widely spaced in time, which would mitigate computational delay issues.

- Data association: Multiple measurements may associate to the same target when, in fact, the measurements were generated as a result of the behavior of different targets. This would be an especially notable issue with the eventual break-up of an iceberg: either a cluster of measurements would be generated or a single measurement would be generated, based on the amount of time that has passed since the breakup. If the cluster was generated from new icebergs that were still relatively close together, then an association error could take place; *i.e.*, a measurement could be associated with an incorrect track. This is an issue where an MHT excels: hypotheses would be generated for each of the potential measurement assignments.

Other composite tracking methods could be considered, such as a solution that uses several multiple interacting multiple model (IMM) filters [89], but it could be difficult to separate targets from one another as a result of their overall similarity.

This research does not address the inherent issues of robotic navigation and sensor data collection in regions of high iceberg density for unmanned vehicles that operate directly on the water; *i.e.*, the types of issues that must be considered when constructing robots that must navigate in these types of harsh and unpredictable conditions. A significant amount of work has been done in these areas by researchers, in particular for AUVs [92–96], but there is still an extensive amount of future work that can be done in this area.

Complete agent drop-out as the result of loss of power is a problem that is also of concern. This is addressed for a centralized arbiter with arbiter hand-off in the description of the centralized sensor reallocation methodology used in this research in Chapter 4, but not addressed for other agents. There are several methods of addressing this issue; a few

examples are as follows:

1. Continue scanning the currently allocated regions until a new allocation is required.
2. Immediately reallocate the agents to new regions based on the current model.
3. Reallocate the agents to the default sensor allocation, and start the modeling process over.

Which one of the many approaches would be considered the best would depend on several factors; *e.g.*, the nature of the environment around the region of interest, proximity to a base station, or the need for agent recovery.

Furthermore, the studies conducted used agents that were mostly homogeneous in capabilities and strictly limited in number; the main differences were in the search patterns used and the capabilities of the sensors. Heterogeneity was demonstrated in the full-scale simulation in Section 6.4, with different agents assigned different roles through the search patterns that were used. Further improvements in performance could be enabled by exploiting agent heterogeneity and having a pool of agents that could adapt in both pattern and number over time.

Additionally, novel ways of transmitting notifications of detected icebergs could be an interesting area of research. For example, a method could be developed that leverages existing social media platforms such as Twitter to provide transmissions of detections/sightings to organizations such as the International Ice Patrol. While the intent of this research is to provide situational awareness for local operations, transmitting the results further up the chain of awareness would be beneficial.

The concept of the ablating source is also not restricted solely to the ice ablation processes that result in the generation of icebergs. Other examples include herds of animals that are separating over time, wreckage deposited in a trail behind a damaged vehicle, and other geophysical processes that resemble ice ablation, such as the movement of rock (albeit an extremely slow ablation process). The approaches in this research can either be

directly applied or modified to apply to these scenarios, which provides for extensive additional research in terms of a) identifying which processes can be represented using ablating sources and b) how to model the generation of new targets, whether by directly applying the Gaussian mixture-based approach in this research or by developing a different modeling approach.

One final research topic would be determining what type of sensor load-out would really be necessary for such a remote sensing task, and what type of vehicle would be best. For the former, examples of such sensors would be: small radars, different electro-optical sensors (*e.g.*, infrared sensors), or video cameras coupled with computer vision algorithms. If necessary, the measurements obtained by these sensors can be fused to produce better estimates, which can then be fed to the modeling algorithms. As for the case of the vehicles, no assumptions have been made about whether they should be unmanned aerial vehicles (UAVs) or autonomous underwater vehicles (AUVs). Not making assumptions about the type of robotic platform allows for a more general study of the problem. However, both types of vehicles have their own advantages and disadvantages. For example, an AUV can be subjected to strong ocean currents, requiring more energy to maintain a consistent search pattern, in addition to being mindful of how far downward an iceberg penetrates into the water. The cost function used to perform target assignment would have to take iceberg depth and ocean current magnitude into account to make superior assignments. Depth or specific ocean currents of any kind are not currently being taken into account by the cost function or the metrics extracted from the iceberg model; ocean currents are effectively rolled into the estimates of iceberg velocity. These specific issues with observing directly at water-level seem to make the case that target observation via UAV might be the better choice. If detailed analysis of iceberg structure is not necessary; *i.e.*, shape, surface size, and position are sufficient, then *in-situ* aerial observation can provide the types of essential measurements required to construct and interpret the iceberg model.

This research has provided a theoretical and practical framework for *in-situ* modeling

of iceberg behavior to allow for determination of model-based search regions for robotic iceberg observation, extracting the useful results that can be produced by that model, and reallocating search resources for more efficient observation of ablating target sources. This contribution allows the field robotics and maritime communities to have an additional perspective on improving observation of these types of targets and target sources, so that operations that are conducted within arctic regions are made safer. In addition to helping maintain safety on the sea, this work can allow for scientific data to be collected in a more efficient manner in a localized region, useful for those scientists that do their work studying the Earth's cryosphere. Icebergs will always be a threat to those who travel across the northern and southern seas, and with the methodologies developed here, they can be a little less threatening.

REFERENCES

[1] S. Savage, "Aspects of iceberg deterioration and drift," in *Geomorphological Fluid Mechanics* (N. Balmforth and A. Provenzale, eds.), vol. 582 of *Lecture Notes in Physics*, pp. 279–318, Springer Berlin Heidelberg, 2001.

[2] M. S. Ramsey, "International Ice Patrol," *International Law & Relations*, vol. 2, p. 55, 1931.

[3] S. Churchill, "Sensor and data fusion of remotely sensed wide-area geospatial targets," Master's thesis, St. John's Newfoundland Canada, January 2009.

[4] D. Diemand, "Icebergs," in *Encyclopedia of Ocean Sciences* (J. H. Steele, K. K. Turekian, and S. A. Thorpe, eds.), Elsevier, 2008.

[5] D. Benn, C. Warren, and R. Mottram, "Calving processes and the dynamics of calving glaciers," *Earth-Science Reviews*, vol. 82, no. 3, pp. 143–179, 2007.

[6] J. A. Dowdeswell, T. J. Benham, T. Strozzi, and J. O. Hagen, "Iceberg calving flux and mass balance of the Austfonna ice cap on Nordaustlandet, Svalbard," *Journal of Geophysical Research*, vol. 113, no. F3, p. F03022, 2008.

[7] G. Moholdt, J. Hagen, T. Eiken, and T. Schuler, "Geometric changes and mass balance of the Austfonna ice cap, Svalbard," *The Cryosphere*, vol. 4, no. 1, pp. 21–34, 2010.

[8] S. Bevan, A. Luckman, T. Murray, H. Sykes, and J. Kohler, "Positive mass balance during the late 20th century on Austfonna, Svalbard, revealed using satellite radar interferometry," *Annals of Glaciology*, vol. 46, no. 1, pp. 117–122, 2007.

[9] T. Dunse, T. V. Schuler, J. O. Hagen, T. Eiken, O. Brandt, and K. A. Hogda, "Recent fluctuations in the extent of the firn area of Austfonna, Svalbard, inferred from GPR," *Annals of Glaciology*, vol. 50, no. 50, pp. 155–162, 2009.

[10] T. E. Vinje, "Some satellite-tracked iceberg drifts in the Antarctic," *Annals of Glaciology*, vol. 1, no. 1, p. 980, 1980.

[11] M. Schodlok, H. Hellmer, G. Rohardt, and E. Fahrbach, "Weddell Sea iceberg drift: Five years of observations," *Journal of Geophysical Research*, vol. 111, no. C6, p. C06018, 2006.

[12] J. L. Lieser, *A Numerical Model for Short-term Sea Ice Forecasting in the Arctic*. PhD thesis, Universität Bremen, 2004.

[13] W. Lahoz, B. Khattatov, and R. Ménard, eds., *Data Assimilation: Making Sense of Observations*. London, England: Springer, 2010.

[14] K. A. Scott, M. Buehner, and T. Carrieres, "An Assessment of Sea-Ice Thickness Along the Labrador Coast from AMSR-E and MODIS Data for Operational Data Assimilation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 5, pp. 2726–2737, 2014.

[15] P. Mathiot, C. K. Beatty, T. Fichefet, H. Goosse, F. Massonnet, and M. Vancoppenolle, "Better constraints on the sea-ice state using global sea-ice data assimilation," *Geoscientific Model Development*, vol. 5, no. 6, pp. 1501–1515, 2012.

[16] A. Caya, M. Buehner, and T. Carrieres, "Analysis and forecasting of sea ice conditions with three-dimensional variational data assimilation and a coupled ice-ocean model," *Journal of Atmospheric and Oceanic Technology*, vol. 27, no. 2, pp. 353–369, 2010.

[17] L. Langebrake, C. Lembke, R. Weisberg, R. Byrne, D. Russell, G. Tilbury, and R. Carr, "Design and initial results of a bottom stationing ocean profiler," in *OCEANS'02 MTS/IEEE*, vol. 1, pp. 98–103, IEEE, 2002.

[18] G. Podnar, J. Dolan, K. Low, and A. Elfes, "Telesupervised remote surface water quality sensing," in *Proceedings of the 2010 IEEE Aerospace Conference*, pp. 1–9, IEEE, 2010.

[19] G. Podnar, J. Dolan, A. Elfes, M. Bergerman, H. Brown, and A. Guisewite, "Human telesupervision of a fleet of autonomous robots for safe and efficient space exploration," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pp. 325–326, ACM, 2006.

[20] M. Dunbabin and A. Grinham, "Experimental evaluation of an Autonomous Surface Vehicle for water quality and greenhouse gas emission monitoring," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5268–5274, IEEE, 2010.

[21] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[22] J. M. Dolan, G. W. Podnar, S. Stancliff, K. H. Low, A. Elfes, J. Higinbotham, J. Hosler, T. Moisan, and J. Moisan, "Cooperative aquatic sensing using the telesupervised adaptive ocean sensor fleet," in *SPIE Europe Remote Sensing*, pp. 747307–747307, International Society for Optics and Photonics, 2009.

[23] G. Podnar, J. Dolan, and A. Elfes, "Networked architecture for robotic environmental ocean science sensors," in *Proceedings of the Workshop: Sensing a Changing World*, 2008.

[24] G. Podnar, J. Dolan, A. Elfes, S. Stancliff, E. Lin, J. Hosier, T. Ames, J. Moisan, T. Moisan, J. Higinbotham, *et al.*, "Operation of robotic science boats using the telesupervised adaptive ocean sensor fleet system," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1061–1068, IEEE, 2008.

[25] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[26] J. Scholtz, J. Young, J. Drury, and H. Yanco, "Evaluation of human-robot interaction awareness in search and rescue," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 2327–2332, IEEE, 2004.

[27] J. Blitch, "Artificial intelligence technologies for robot assisted urban search and rescue," *Expert Systems with Applications*, vol. 11, no. 2, pp. 109–124, 1996.

[28] I. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human-robot teaming for search and rescue," *Pervasive Computing, IEEE*, vol. 4, no. 1, pp. 72–79, 2005.

[29] G. Cannata and A. Sgorbissa, "A minimalist algorithm for multirobot continuous coverage," *IEEE Transactions on Robotics*, vol. 27, pp. 297–312, Apr 2011.

[30] L. Parker and B. Emmons, "Cooperative multi-robot observation of multiple moving targets," in *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2082–2089, IEEE, 1997.

[31] L. Parker, "Distributed algorithms for multi-robot observation of multiple moving targets," *Autonomous Robots*, vol. 12, no. 3, pp. 231–255, 2002.

[32] L. Parker and C. Touzet, "Multi-robot learning in a cooperative observation task," *Distributed Autonomous Robotic Systems*, vol. 4, pp. 391–401, 2000.

[33] B. Jung and G. Sukhatme, "Tracking targets using multiple robots: The effect of environment occlusion," *Autonomous Robots*, vol. 13, no. 3, pp. 191–205, 2002.

[34] B. Jung and G. Sukhatme, "A generalized region-based approach for multi-target tracking in outdoor environments," in *Proceedings of the 2004 International Conference on Robotics and Automation*, vol. 3, pp. 2189–2195, IEEE, 2004.

[35] S. Luke, K. Sullivan, L. Panait, and G. Balan, "Tunably decentralized algorithms for cooperative target observation," in *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 911–917, ACM, 2005.

[36] A. Kolling and S. Carpin, "Multirobot cooperation for surveillance of multiple moving targets-a new behavioral approach," in *Proceedings of the 2006 International Conference on Robotics and Automation*, pp. 1311–1316, IEEE, 2006.

[37] A. Kolling and S. Carpin, "Cooperative observation of multiple moving targets: an algorithm and its formalization," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 935–953, 2007.

[38] J. Kuhn, C. Reinl, and O. von Stryk, "Predictive control for multi-robot observation of multiple moving targets based on discrete-continuous linear models," in *Proceedings of the 18th IFAC World Congress*, pp. 257–262, 2011.

[39] R. Hashemi, L. Jin, G. Anderson, E. Wilson, and M. Clark, "A comparison of search patterns for cooperative robots operating in remote environment," in *Proceedings of the International Conference on Information Technology: Coding and Computing, 2001*, pp. 668–672, IEEE, 2001.

[40] E. Tunstel, J. Dolan, T. Fong, and D. Schreckenghost, "Mobile robotic surveying performance for planetary surface site characterization," in *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, pp. 200–205, ACM, 2008.

[41] L. T. Parker, R. A. Coogle, and A. M. Howard, "Estimation-informed, resource-aware robot navigation for environmental monitoring applications," in *Proceedings of the 2013 International Conference on Robotics and Automation*, vol. 1, pp. 1033–1038, IEEE, 2013.

[42] B. Everitt and D. Hand, *Finite Mixture Distributions*. London; New York: Chapman and Hall, 1981.

[43] M. R. Gupta and Y. Chen, "Theory and use of the EM algorithm," *Foundations and Trends in Signal Processing*, vol. 4, no. 3, pp. 223–296, 2010.

[44] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, University of California, 1967.

[45] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, pp. 716–723, dec 1974.

[46] C. M. Hurvich and C.-L. Tsai, "Regression and time series model selection in small samples," *Biometrika*, vol. 76, pp. 297–307, 1989.

[47] D. J. Salmond, "Mixture reduction algorithms for target tracking in clutter," in *SPIE signal and data processing of small targets*, vol. 1305, pp. 434–445, 1990.

[48] D. Salmond, "Mixture reduction algorithms for point and extended object tracking in clutter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 2, pp. 667–686, 2009.

[49] J. L. Williams and P. S. Maybeck, "Cost-function-based Gaussian mixture reduction for target tracking," in *Proceedings of the Sixth International Conference of Information Fusion*, vol. 2, pp. 1047–1054, 2003.

[50] A. R. Runnalls, "Kullback-Leibler approach to Gaussian mixture reduction," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, pp. 989–999, 2007.

[51] M. F. Huber and U. D. Hanebeck, "Progressive Gaussian mixture reduction," in *2008 11th International Conference on Information Fusion*, pp. 1–8, IEEE, 2008.

[52] T. Kailath, "The Divergence and Bhattacharyya Distance Measures in Signal Selection," *IEEE Transactions on Communication Technology*, vol. 15, pp. 52–60, Feb. 1967.

[53] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2000*, vol. 2, pp. 142–149, 2000.

[54] "International Ice Patrol (IIP) sightings database." Available Online: http://www.navcen.uscg.gov/?pageName=IIPHome.

[55] R. A. Coogle and A. M. Howard, "Modeling and Metrics for In-Situ Iceberg Activity Monitoring." Submitted for publication.

[56] C. F. F. Karney, "Transverse mercator with an accuracy of a few nanometers," *Journal of Geodesy*, vol. 85, no. 8, pp. 475–485, 2011.

[57] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tessellations: applications and algorithms," *SIAM review*, vol. 41, no. 4, pp. 637–676, 1999.

[58] R. A. Coogle and A. M. Howard, "Robotic Resource Allocation for the Observation of Ablating Target Sources." To be presented at the 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC).

[59] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[60] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial & Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[61] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, 1987.

[62] D. P. Bertsekas, *Linear Network Optimization*. Cambridge: MIT Press, 1991.

[63] G. L. Nemhauser and L. A. Wolsey, *Integer and combinatorial optimization*, vol. 18. Wiley New York, 1988.

[64] M. Dias and A. Stentz, "Opportunistic optimization for market-based multirobot control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002*, vol. 3, pp. 2714–2720, IEEE, 2002.

[65] B. Gerkey and M. Matarić, "Sold!: Auction methods for multirobot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.

[66] A. Viguria and A. Howard, "An integrated approach for achieving multirobot task formations," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 176–186, 2009.

[67] M. Pavone and E. Frazzoli, "Decentralized policies for geometric pattern formation and path coverage," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, p. 633, 2007.

[68] N. Michael, M. Zavlanos, V. Kumar, and G. Pappas, "Distributed multi-robot task assignment and formation control," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pp. 128–133, May 2008.

[69] K. Lerman, C. Jones, A. Galstyan, and M. J. Matarić, "Analysis of dynamic task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 25, no. 3, pp. 225–241, 2006.

[70] A. Viguria and A. M. Howard, "Probabilistic analysis of market-based algorithms for initial robotic formations," *The International Journal of Robotics Research*, vol. 29, no. 9, pp. 1154–1172, 2010.

[71] A. Viguria, I. Maza, and A. Ollero, "S+T: An algorithm for distributed multirobot task allocation based on services for improving robot cooperation," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pp. 3163–3168, May 2008.

[72] K. Alexis and A. Tzes, "Application of a quadtree-based market-allocation algorithm for cellular environment coverage with cooperative robots," in *16th Mediterranean Conference on Control and Automation*, pp. 938–945, June 2008.

[73] P. Amstutz, N. Correll, and A. Martinoli, "Distributed boundary coverage with a team of networked miniature robots using a robust market-based algorithm," *Annals of Mathematics and Artificial Intelligence*, vol. 52, pp. 307–333, 2008. 10.1007/s10472-009-9127-8.

[74] F. Tang and L. Parker, "A complete methodology for generating multi-robot task solutions using ASyMTRe-D and market-based task allocation," in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pp. 3351–3358, April 2007.

[75] L. Lin, W. Lei, Z. Zheng, and Z. Sun, "A learning market based layered multi-robot architecture," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3417–3422, May 2004.

[76] M. Dias, B. Ghanem, and A. Stentz, "Improving cost estimation in market-based coordination of a distributed sensing task," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3972–3977, Aug 2005.

[77] H. Seraji and A. Howard, "Behavior-based robot navigation on challenging terrain: A fuzzy logic approach," *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 308–321, June 2002.

[78] S. Williams and A. Howard, "Developing monocular visual pose estimation for arctic environments," *Journal of Field Robotics*, vol. 27, no. 2, pp. 145–157, 2010.

[79] E. Olson, J. Leonard, and S. Teller, "Robust range-only beacon localization," in *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pp. 66–75, June 2004.

[80] M. Grewal, L. Weill, and A. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*. New York: Wiley, 2007.

[81] J. Lee, R. Huang, A. Vaughn, X. Xiao, J. K. Hedrick, M. Zennaro, and R. Sengupta, "Strategies of path-planning for a UAV to track a ground vehicle," in *AINS Conference*, vol. 2003, 2003.

[82] L. Thomas, S. T. Buckland, K. P. Burnham, D. R. Anderson, J. L. Laake, D. L. Borchers, and S. Strindberg, *Distance Sampling*. John Wiley & Sons, Ltd, 2006.

[83] J. D. Lawrence, *A Catalog of Special Plane Curves*. New York: Dover Publications, 1972.

[84] B. Gerkey, R. Vaughan, and A. Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," in *Proceedings of the International Conference on Advanced Robotics*, pp. 317–323, IEEE, 2003.

[85] R. A. Coogle and A. M. Howard, "The Iceberg Observation Problem: Using Multiple Agents to Monitor and Observe Ablating Target Sources," in *Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, vol. 1, pp. 1–6, IEEE, 2013.

[86] M. Skolnik, *Radar Handbook*. New York: McGraw-Hill, 2nd ed., 1990.

[87] R. A. Coogle and A. M. Howard, "A Multiagent Robotic System for In-Situ Modeling and Observation of Icebergs," December 2013. Presented at the 2013 American Geophysical Union Fall Meeting.

[88] C. Kunz, C. Murphy, R. Camilli, H. Singh, J. Bailey, R. Eustice, M. Jakuba, K.-i. Nakamura, C. Roman, T. Sato, *et al.*, "Deep sea underwater robotic exploration in the ice-covered Arctic ocean with AUVs," in *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3654–3660, IEEE, 2008.

[89] Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: John Wiley and Sons, 2001.

[90] S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[91] R. L. Streit and T. E. Luginbuhl, "Maximum likelihood method for probabilistic multihypothesis tracking," in *SPIE's International Symposium on Optical Engineering and Photonics in Aerospace Sensing*, pp. 394–405, International Society for Optics and Photonics, 1994.

[92] P. Kimball and S. Rock, "Estimation of iceberg motion for mapping by AUVs," in *Proceedings of the 2010 IEEE/OES Conference on Autonomous Underwater Vehicles (AUV)*, pp. 1–9, IEEE, 2010.

[93] P. Kimball and S. Rock, "Sonar-based iceberg-relative AUV navigation," in *Proceedings of the 2008 IEEE/OES Conference on Autonomous Underwater Vehicles*, pp. 1–6, IEEE, 2008.

[94] J. Bellingham, C. Goudey, T. Consi, J. Bales, D. Atwood, J. Leonard, and C. Chryssostomidis, "A second generation survey AUV," in *Proceedings of the 1994 Symposium on Autonomous Underwater Vehicle Technology*, pp. 148–155, IEEE, 1994.

[95] R. McEwen, H. Thomas, D. Weber, and F. Psota, "Performance of an AUV navigation system at Arctic latitudes," *IEEE Journal of Oceanic Engineering*, vol. 30, no. 2, pp. 443–454, 2005.

[96] K. Richmond, S. Gulati, C. Flesher, B. P. Hogan, and W. C. Stone, "Navigation, control, and recovery of the ENDURANCE under-ice hovering AUV," in *UUST: International Symposium on Unmanned Untethered Submersible Technology* (R. Blidberg, ed.), (Durham, NH), Autonomous Undersea Systems Institute, Aug 2009.